Chapter 5

Two-Dimensional Geometric Transformations

➢ Main Content

- 5.1 Basic Two-Dimensional Geometric Transformations
- 5.2 Matrix Representations and Homogeneous Coordinates
- 5.3 Inverse Transformations
- 5.4 Two-Dimensional Composite Transformations
- 5.5 Other Two-Dimensional Transformations
- 5.6 Raster Methods for Geometric Transformations
- 5.7 Transformation between Two-Dimensional Coordinate Systems

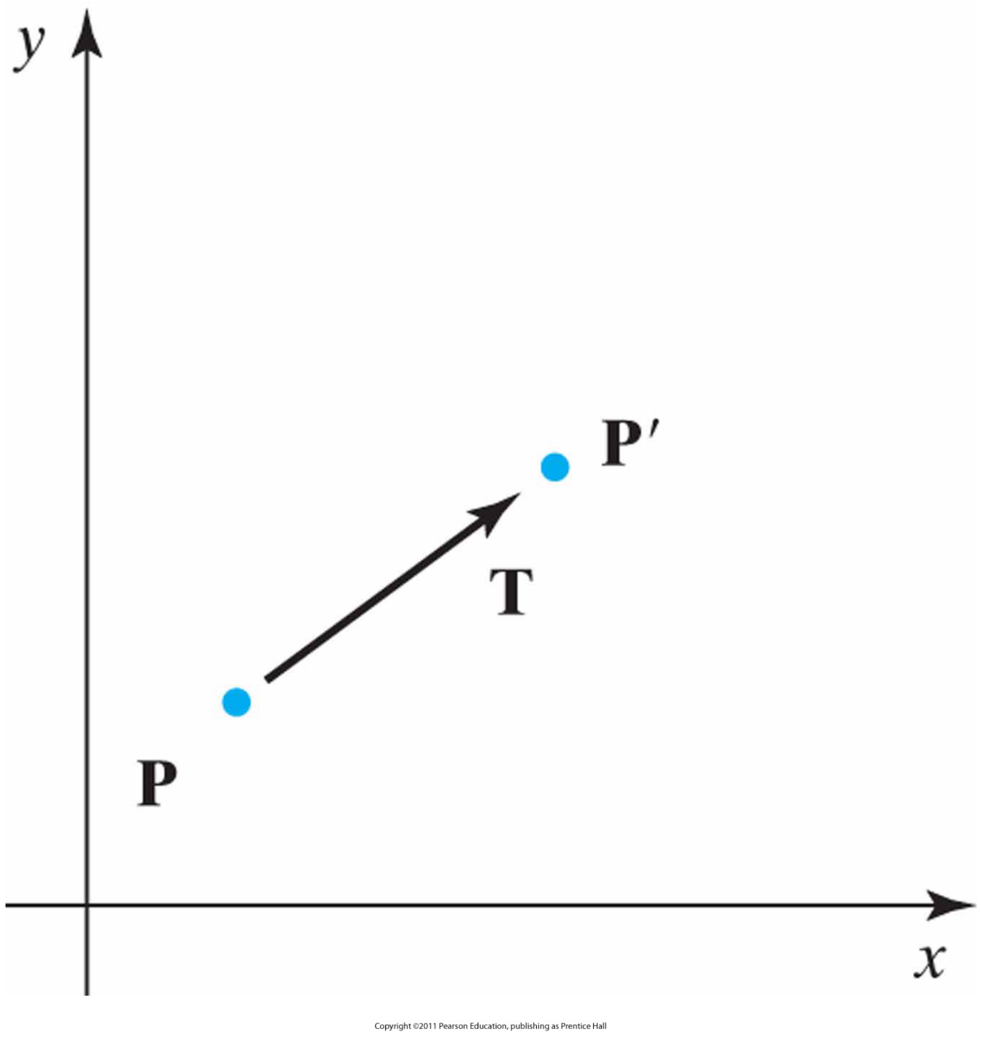
5.1 Basic Two-Dimensional Geometric Transformations

- **In all graphics:**
  - Translation
  - Rotation
  - Scaling
- **In some packages:**
  - Reflection Operation
  - Shearing Operation

- **Two-Dimensional Geometric Translation**

  - We perform a translation on a single coordinate point by adding offsets to its coordinate so as to generate a new coordinate position.

*Translating a point from position **P** to position **P′** using a translation vector **T***



*Add **translation distances** t_x and t_y to the original point (x, y) to obtain the new coordinate position (x', y')*

$$x' = x + t_x \qquad y' = \mathrm{y} + t_y$$

*(t_x, t_y) is called a **translation vector** or **shift vector***

$$\mathbf{P} = \begin{bmatrix} x \\ y \end{bmatrix} \quad \mathbf{P}' = \begin{bmatrix} x' \\ y' \end{bmatrix} \quad \mathbf{T} = \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

Matrix form

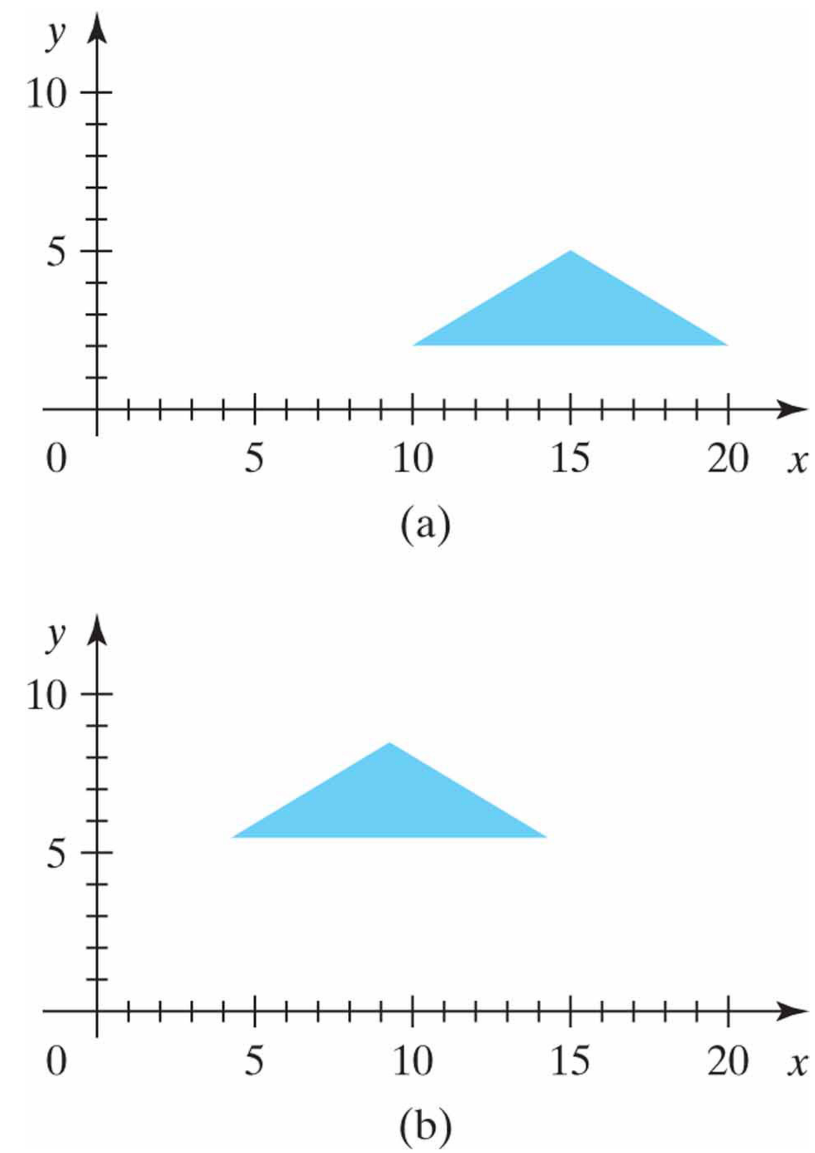
$$\mathbf{P}' = \mathbf{P} + \mathbf{T}$$

- **Two-Dimensional Geometric Translation**

$$\boxed{\mathbf{P}' = \mathbf{P} + \mathbf{T}}$$

➢ Translation is a *rigid-body* transformation that moves objects without deformation.

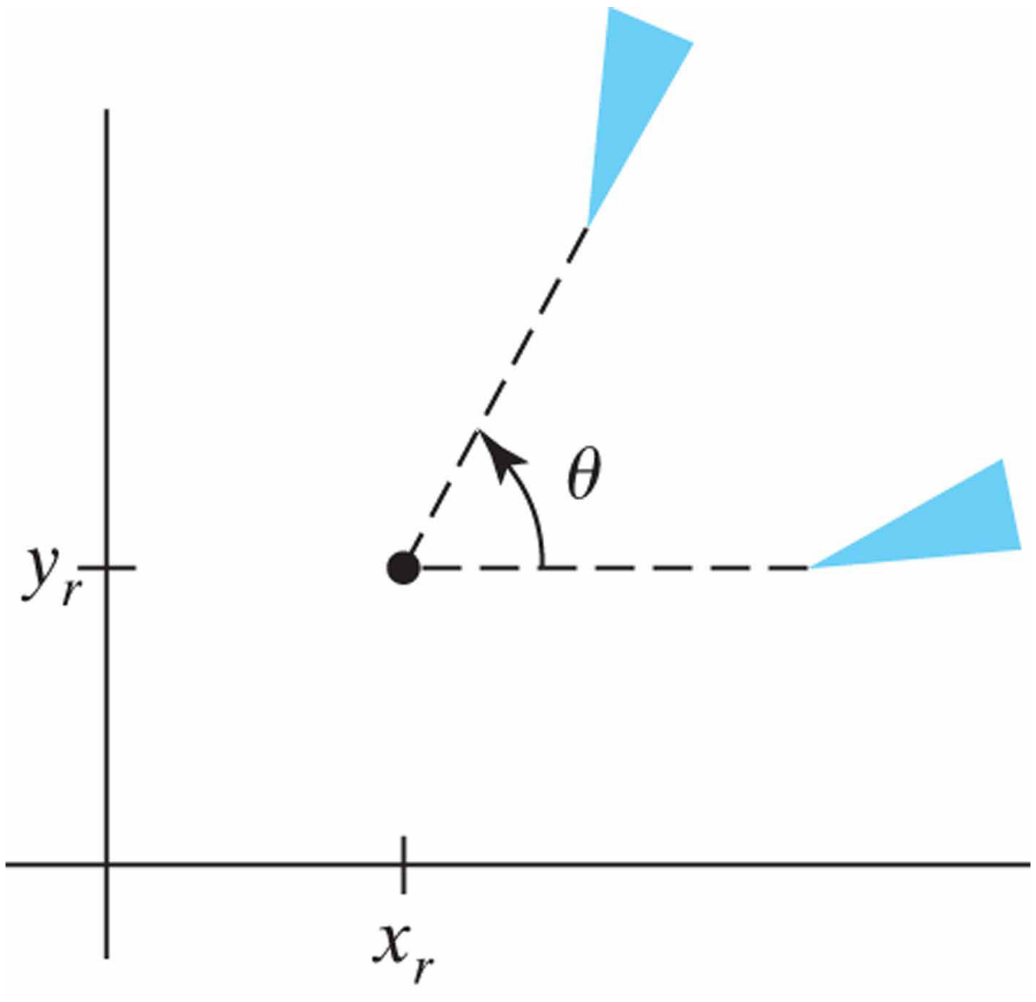
- *A straight-ling segment s translated by applying to each of endpoints and redrawing the line between the new endpoint positions.*
- *We add a translation vector to the coordinate position of each vertex and then regenerate the polygon using the new set of vertex coordinates.*



- **Two-Dimensional Rotation**

  - A two-dimensional rotation of an object is obtained by repositioning the object along a circular path in the *xy* plane.

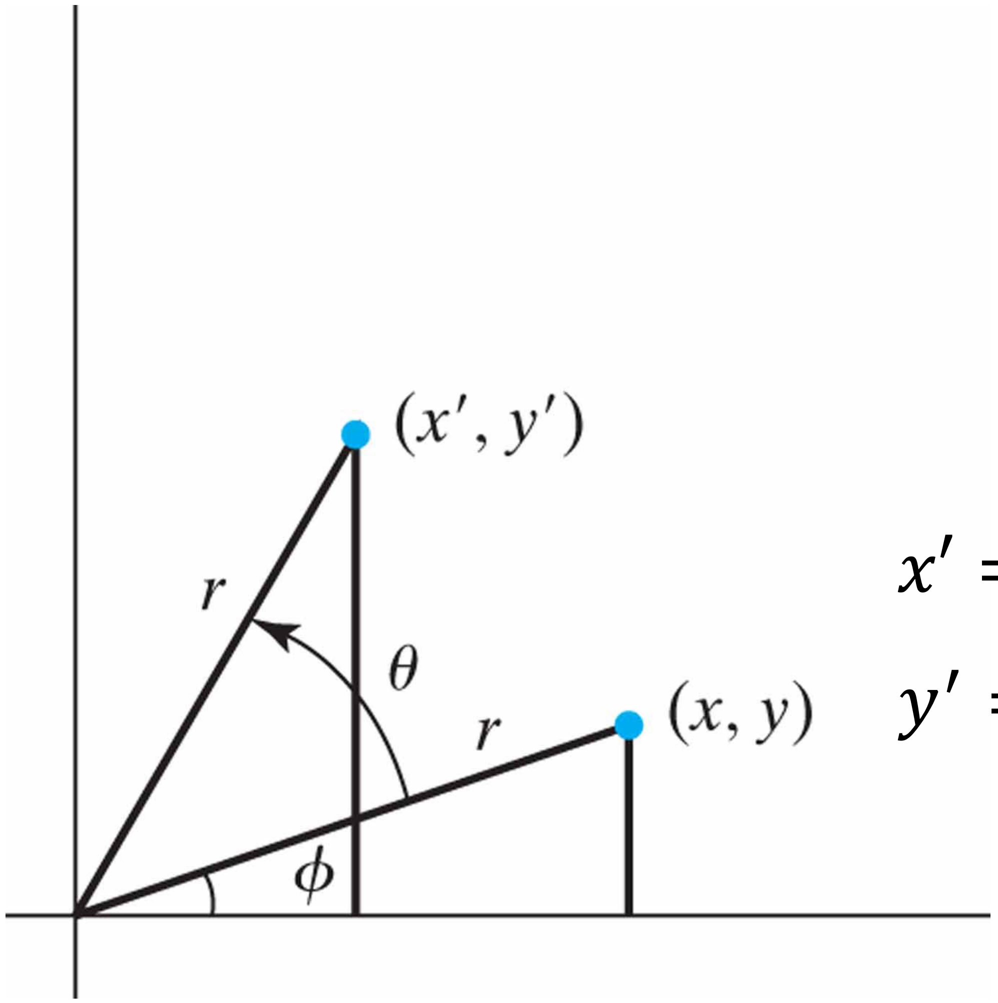
    - *Parameters for the two-dimensional rotation are the rotation angle θ and a position (x_r, y_r), called the* ***rotation point*** *(or* ***pivot point****).*
    - *A positive value for the angle θ defines a counter-clockwise rotation about the pivot point.*



- **Two-Dimensional Rotation**

  - We first determine the transformation equations for rotation of a point position **P** when the pivot point is at the coordinate origin.

Rotation of a point from position (x, y) to position (x', y') through an angle θ relative to the coordinate origin. The original angular displacement of the point from the x axis is Φ



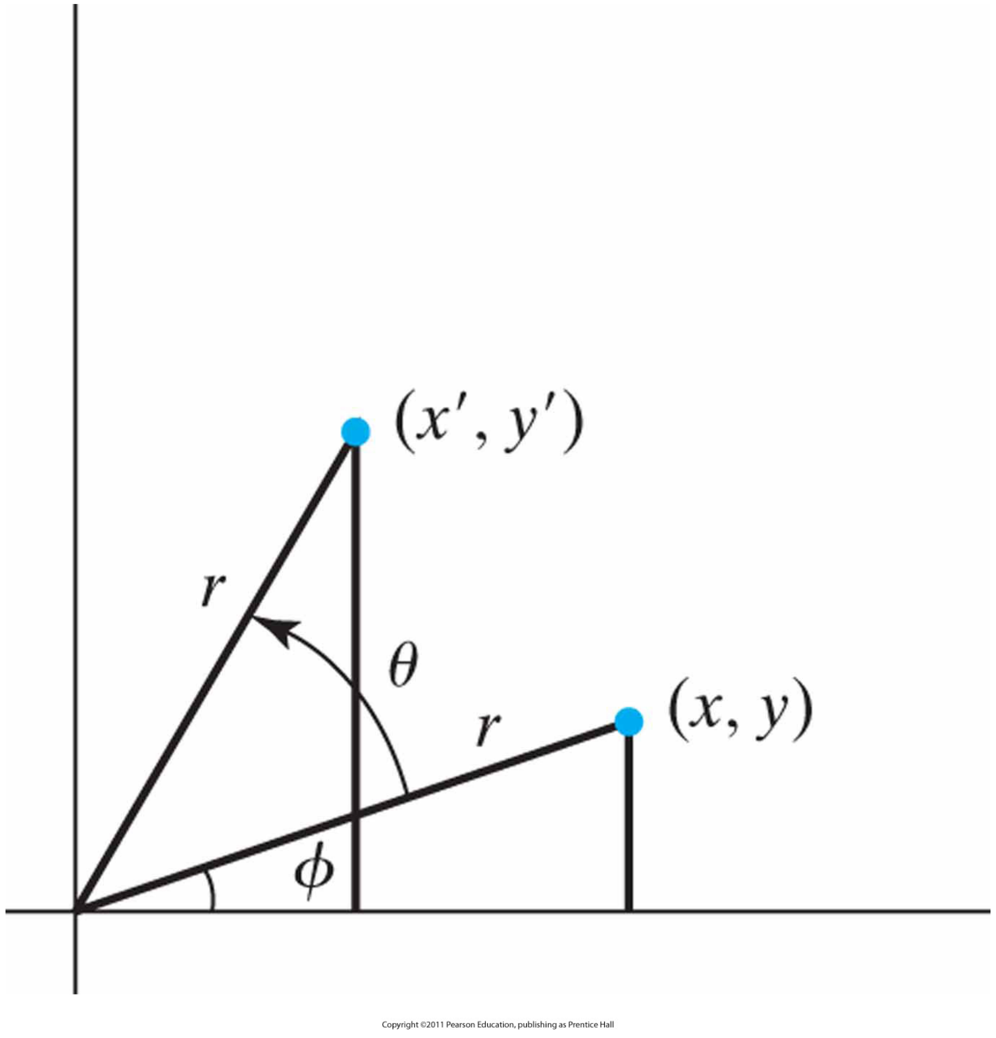
- *Using standard trigonometric identities, we can express the transformed coordinates in terms of angles θ and Φ as*

$$x' = r\cos(\phi+\theta) = r\cos\phi\cos\theta - r\sin\phi\sin\theta$$

$$y' = r\sin(\phi+\theta) = r\cos\phi\sin\theta + r\sin\phi\cos\theta$$

• Two-Dimensional Rotation

$$\boxed{\begin{aligned} x' &= r\cos\phi\cos\theta - r\sin\phi\sin\theta \\ y' &= r\cos\phi\sin\theta + r\sin\phi\cos\theta \end{aligned}}$$



- *The original coordinates of the point in polar coordinates are*

$$x = r\cos\phi \qquad y = r\sin\phi$$

- *The transformation equation for rotating a point at position (x,y) through an angle θ about the origin:*

$$x' = x\cos\theta - y\sin\theta$$

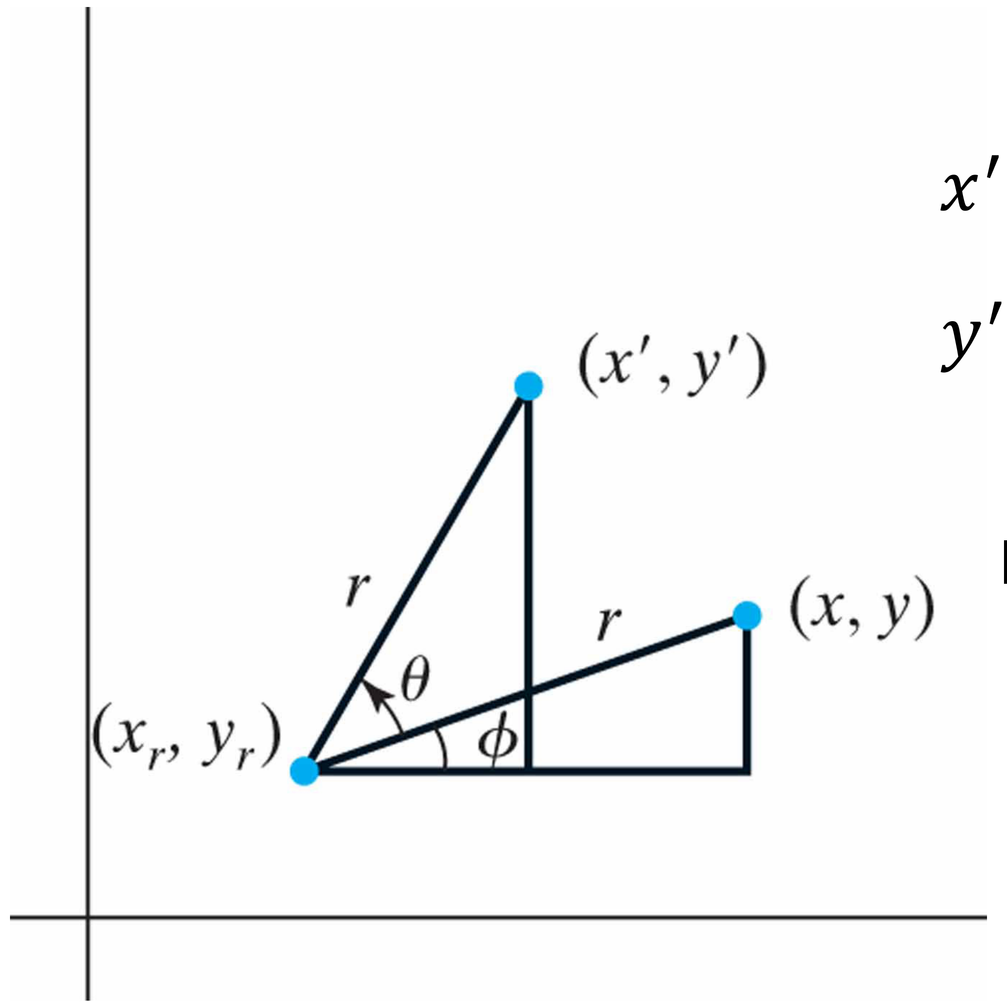
$$y' = x\sin\theta + y\cos\theta$$

- *Matrix form:*

$$\mathbf{P}' = \mathbf{R}\cdot\mathbf{P}$$

$$\mathbf{R} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

- **Two-Dimensional Rotation**
  - Using the trigonometric relationships indicated by the two right triangles, we can obtain the transformation equations for rotation of a point about any specified rotation position (x_r, y_r) :



$$x' = x_r + (x - x_r)\cos\theta - (y - y_r)\sin\theta$$

$$y' = y_r + (x - x_r)\sin\theta + (y - y_r)\cos\theta$$

- *As with translations, rotations are rigid-body transformations that move objects without deformation.*

- **Two-Dimensional Scaling**
  - To alter the size of an object, we apply a **scaling** transformation.

$$x' = x \cdot s_x \quad \mathrm{y}' = \mathrm{y} \cdot s_y$$

where, scaling factor s_x scales an object in the x direction, while s_y scales in the y direction

Matrix form:

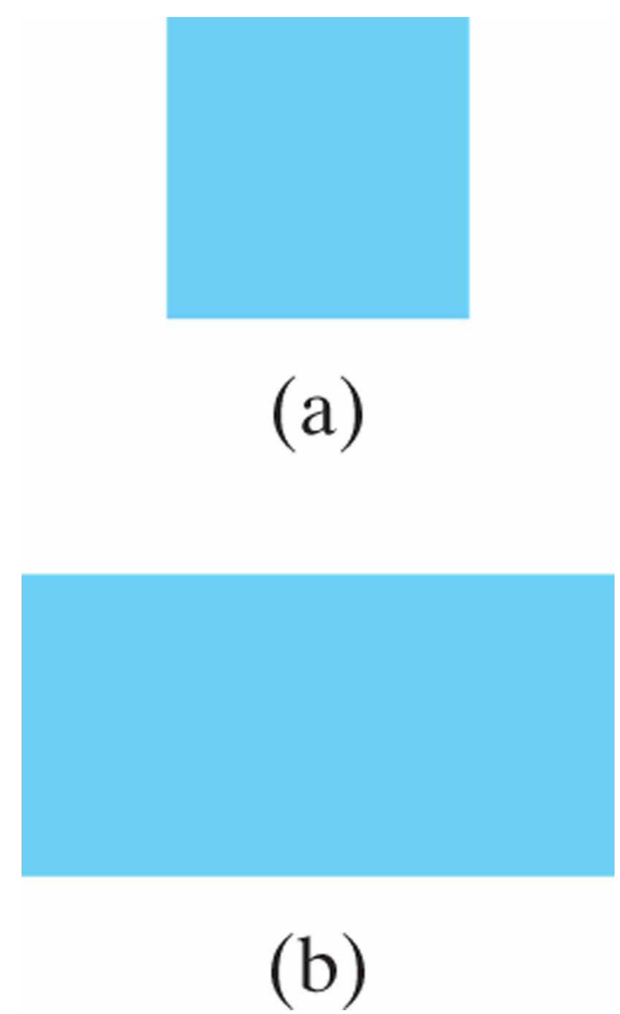
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

Or

$$\mathbf{P}' = \mathbf{S} \cdot \mathbf{P}$$

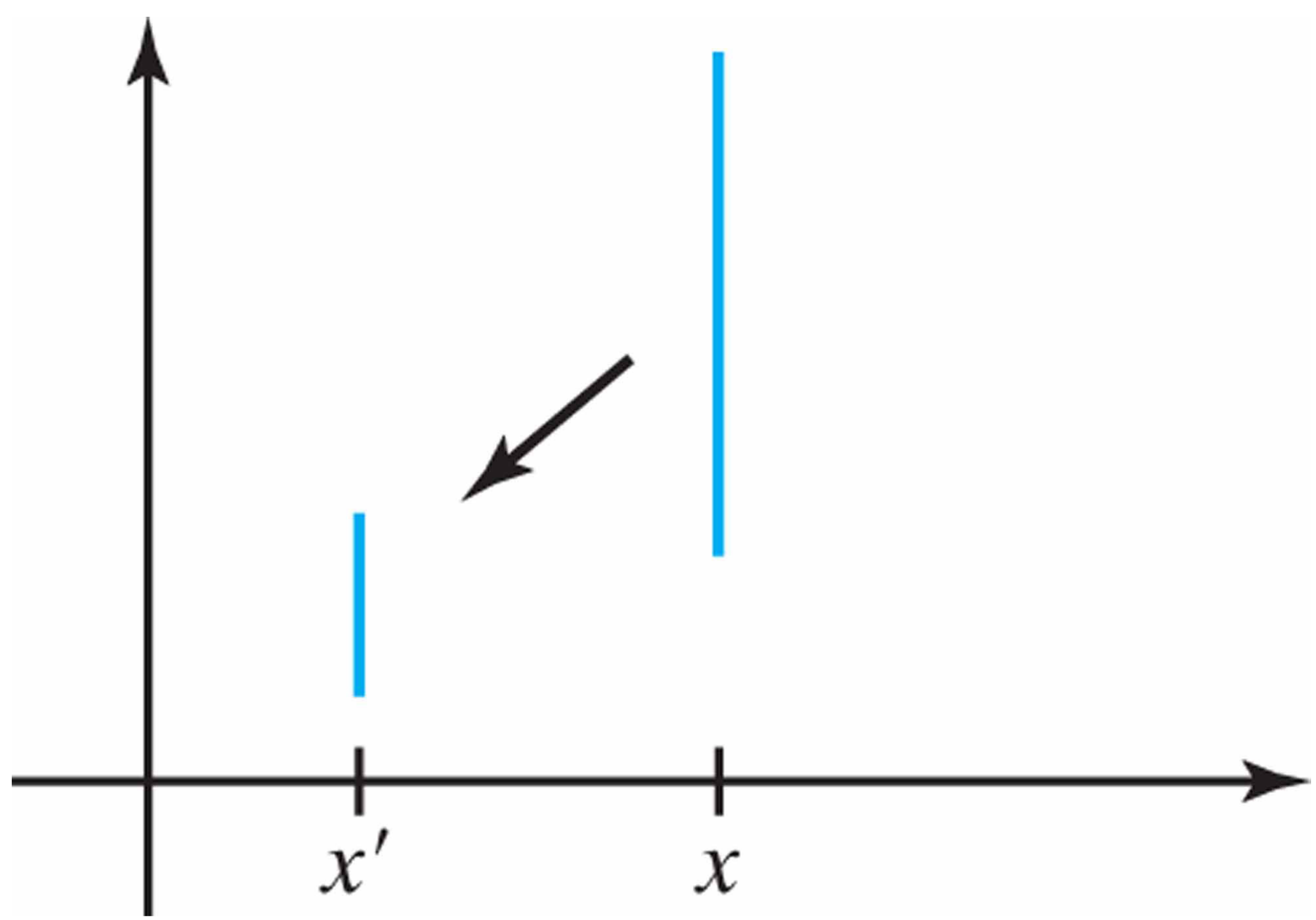
- **Two-Dimensional Scaling**
  - When s_x and s_y are assigned the same value, a **uniform scaling** is produced, which maintains relative object proportions.
  - Unequal values for s_x and s_y result in a **differential scaling** that is often used in design applications.

Turning a square (a) into a rectangle (b) with scaling factors s_x = 2 and s_y = 1



- **Two-Dimensional Scaling**

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow$$ Both scaling and repositioning



- **Two-Dimensional Scaling**
  - We can control the location of a scaled object by choosing a positon, called the fixed point, that is to remain unchanged after the scaling transformation.

For a coordinate position (x, y)*, the scaled coordinates* (x', y') *are then calculated as*

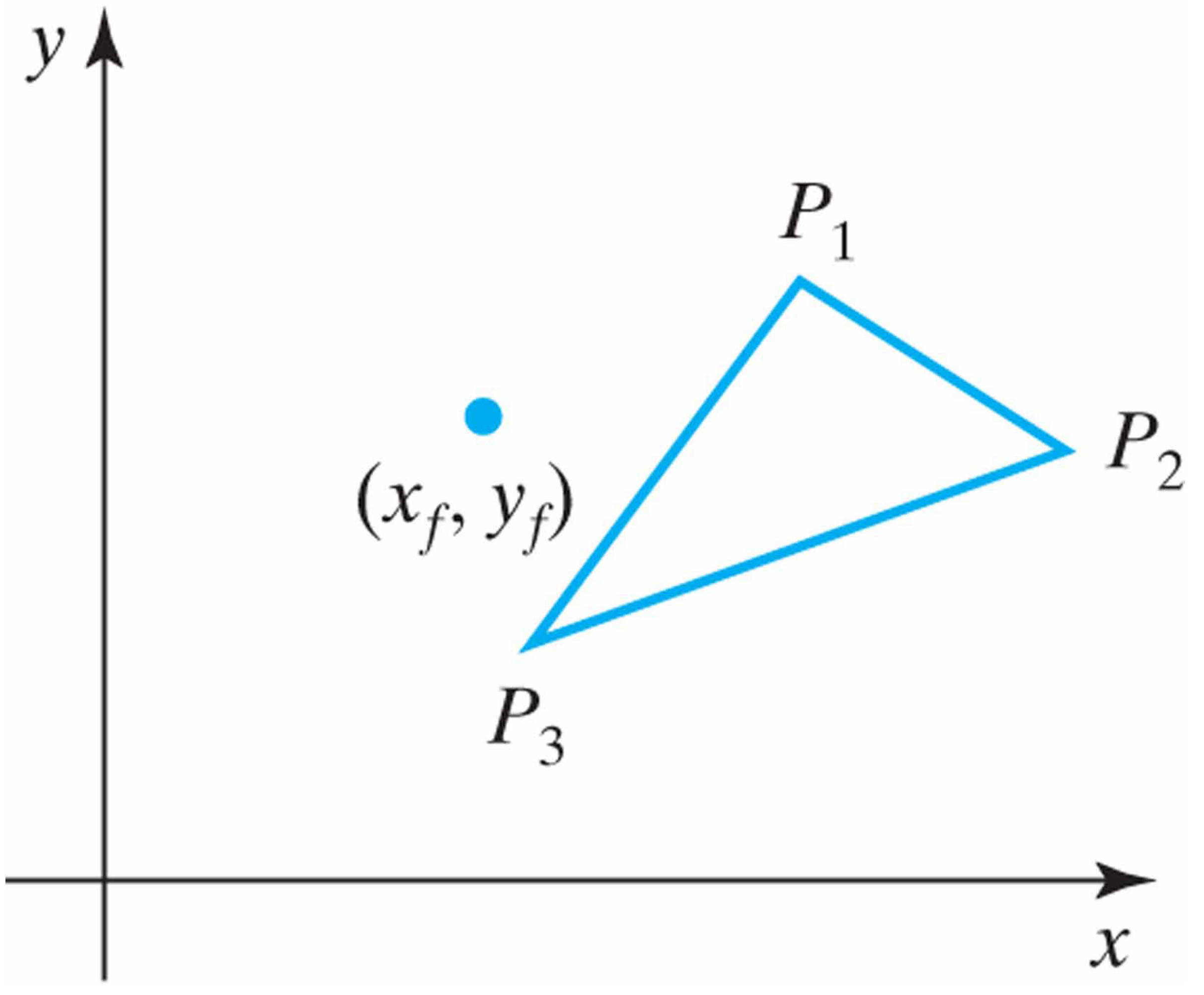
$$x' - x_f = (x - x_f)s_x$$

$$y' - y_f = (y - y_f)s_y$$

Rewrite

$$x' = x \cdot s_x + x_f(1 - s_x)$$

$$y' = y \cdot s_x + y_f(1 - s_y)$$



5.2 Matrix Representations and Homogeneous Coordinates

- Many graphics applications involve sequences of geometric transformations.

- In design and picture construction applications, we perform translations, rotations, and scalings to fit the picture components into their proper positions.

- Matrix representations can be reformulated.

- **Matrix Representations**
  - Each of the three basic two-dimensional transformations (translation, rotation, and scaling) can be expressed in the general matrix form:

$$\mathbf{P}' = \mathbf{M}_1 \cdot \mathbf{P} + \mathbf{M}_2$$

  - With coordinate positions P and P' represented as column vectors. $\mathbf{M}_1$ is a 2x2 array containing multiplicative factors, and $\mathbf{M}_2$ is a two-element column matrix containing translational terms.
  - For translation, $\mathbf{M}_1$ is the identity matrix. For rotation or scaling, $\mathbf{M}_2$ contains the translational terms associated with the pivot point or scaling fixed point.

- **Homogeneous Coordinates**
  - Multiplicative and translational terms for a two-dimensional geometric transformation can be combined into a single matrix if we expand the representations to 3x3 matrices.

$$(x, y) \longrightarrow (x_h, y_h, h)$$

where the homogeneous parameter h is a nonzero value such that

$$x = \frac{x_h}{h} \qquad y = \frac{y_h}{h}$$

  - A convenient choice is simply to set $h = 1$. Each two-dimensional position is then represented with $(x, y, 1)$

- **Two-Dimensional Translation Matrix**

  - We can represent the equations for a two-dimensional translation of a coordinate position:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

  - This translation operation can be written in the abbreviated form.

$$\mathbf{P}' = \mathbf{T}(t_x, t_y) \cdot \mathbf{P}$$

- **Two-Dimensional Rotation Matrix**
  - Similarly, two-dimensional rotation transformation equations about the coordinate origin can be expressed in the matrix form：

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Or as

$$\mathbf{P}' = \mathbf{R}(\theta) \cdot \mathbf{P}$$

- **Two-Dimensional Scaling Matrix**
  - Finally, a scaling transformation relative to the coordinate origin can now be expressed as the matrix multiplication:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Or as

$$\mathbf{P}' = \mathbf{S}(s_x, s_y) \cdot \mathbf{P}$$

5.3 Inverse Transformations

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{T}^{-1} = \begin{bmatrix} 1 & 0 & -t_x \\ 0 & 1 & -t_y \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{R} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{R}^{-1} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{S} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{S}^{-1} = \begin{bmatrix} \frac{1}{s_x} & 0 & 0 \\ 0 & \frac{1}{s_y} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

5.4 Two-Dimensional Composite Transformations

- Forming products of transformation matrices is often referred to as a **concatenation**, or **composition**, of matrices.

$$\mathbf{P}' = \mathbf{M}_2 \cdot \mathbf{M}_1 \cdot \mathbf{P} = \mathbf{M} \cdot \boldsymbol{P}$$

- The coordinate position is transformed using the composite matrix $\mathbf{M}$, rather than applying the individual transformations $\mathbf{M}_1$ and then $\mathbf{M}_2$.

- **Composite Two-Dimensional Translations**
  - Two successive translation vectors $T(t_{x1},t_{y1})$, $T(t_{x2},t_{y2})$ are applied to a two-dimensional coordinate position **P**, the final transformed location **P'**.

$$\mathbf{P}' = \mathbf{T}\left(t_{2x}, t_{2y}\right) \cdot \left\{\mathbf{T}\left(t_{1x}, t_{1y}\right) \cdot \mathbf{P}\right\} = \left\{\mathbf{T}\left(t_{2x}, t_{2y}\right) \cdot \mathbf{T}\left(t_{1x}, t_{1y}\right)\right\} \cdot \mathbf{P}$$

The composite transformation matrix for this sequence of translations is

$$\begin{bmatrix} 1 & 0 & t_{2x} \\ 0 & 1 & t_{2y} \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & t_{1x} \\ 0 & 1 & t_{1y} \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_{1x} + t_{2x} \\ 0 & 1 & t_{1y} + t_{2y} \\ 0 & 0 & 1 \end{bmatrix}$$

Or

$$\mathbf{T}\left(t_{2x}, t_{2y}\right) \cdot \mathbf{T}\left(t_{1x}, t_{1y}\right) = \mathbf{T}\left(t_{1x} + t_{2x}, t_{1y} + t_{2y}\right)$$

- **Composite Two-Dimensional Rotations**

  - ➢ Two successive rotations applied to a point **P** produce the transformed position:

$$\mathbf{P}' = \mathbf{R}(\theta_2) \cdot \{\mathbf{R}(\theta_1) \cdot \mathbf{P}\} = \{\mathbf{R}(\theta_2) \cdot \mathbf{R}(\theta_1)\} \cdot \mathbf{P}$$

  - ➢ We can verify that two successive rotations are additive:

$$\mathbf{R}(\theta_2) \cdot \mathbf{R}(\theta_1) = \mathbf{R}(\theta_1 + \theta_2)$$

  - ➢ So that the final rotated coordinates of a point can be calculated with the composite rotation matrix as:

$$\mathbf{P}' = \mathbf{R}(\theta_1 + \theta_2) \cdot \mathbf{P}$$

- **Composite Two-Dimensional Scalings**

  - Concatenating transformation matrices for two successive scaling operations in two dimensions produces the following composite scaling matrix:

$$\begin{bmatrix} s_{2x} & 0 & 0 \\ 0 & s_{2y} & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} s_{1x} & 0 & 0 \\ 0 & s_{1y} & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} s_{1x} \cdot s_{2x} & 0 & 0 \\ 0 & s_{1y} \cdot s_{2y} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

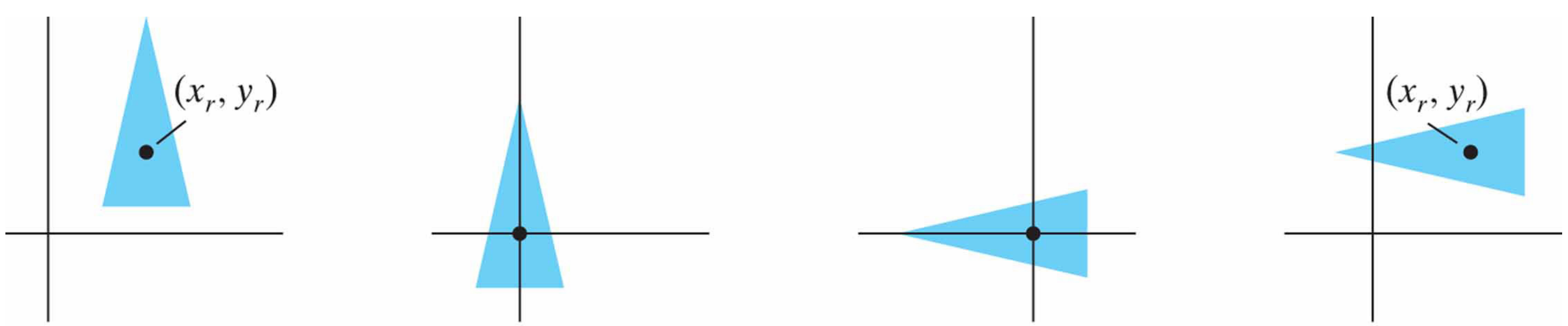
Or

$$\mathbf{S}(s_{2x}, s_{2y}) \cdot \mathbf{S}(s_{1x}, s_{1y}) = \mathbf{S}(s_{1x} \cdot s_{2x}, s_{1y} \cdot s_{2y})$$

  - The resulting matrix in this case indicates that successive scaling operation are **multiplicative**.

- **General Two-Dimensional Pivot-Point Rotation**

  1. Translate the object so that the pivot-point position is moved to the coordinate origin.
  2. Rotate the object about the coordinate origin.
  3. Translate the object so that the pivot point is returned to its original position.



(a) Original Position of Object and Pivot Point

(b) Translation of Object so that Pivot Point (x_r, y_r) is at Origin

(c) Rotation about Origin

(d) Translation of Object so that the Pivot Point is Returned to Position (x_r, y_r)

- **General Two-Dimensional Pivot-Point Rotation**
  - The composite transformation matrix for this sequence is obtained with the concatenation:

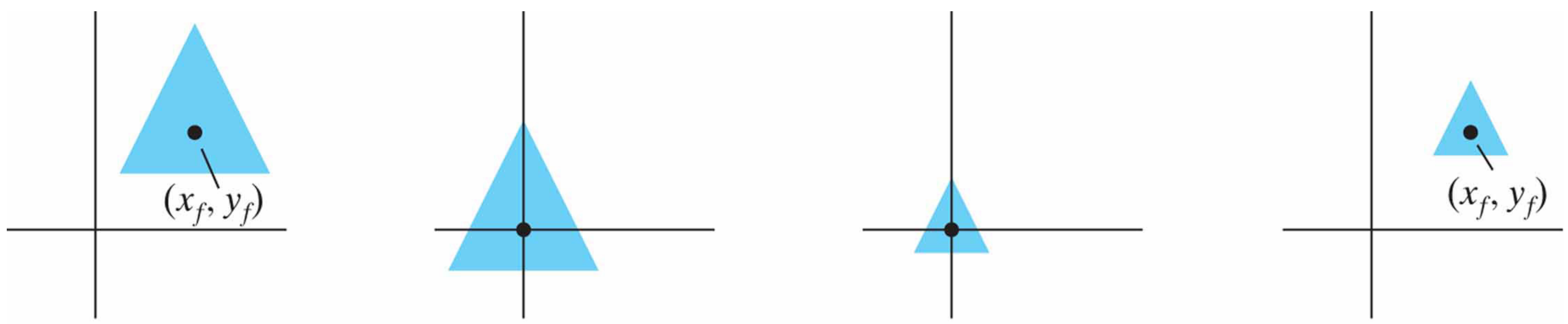
$$\begin{bmatrix} 1 & 0 & x_r \\ 0 & 1 & y_r \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -x_r \\ 0 & 1 & -y_r \\ 0 & 0 & 1 \end{bmatrix}$$
$$= \begin{bmatrix} \cos\theta & -\sin\theta & x_r(1-\cos\theta) + y_r\sin\theta \\ \sin\theta & \cos\theta & y_r(1-\cos\theta) - x_r\sin\theta \\ 0 & 0 & 1 \end{bmatrix}$$

Or

$$\mathbf{T}(x_{\mathrm{r}}, y_{\mathrm{r}}) \cdot \mathbf{R}(\theta) \cdot \mathbf{T}(-x_{\mathrm{r}}, -y_{\mathrm{r}}) = \mathbf{R}(x_{\mathrm{r}}, y_{\mathrm{r}}, \theta)$$

- **General Two-Dimensional Fixed-Point Scaling**

  1. Translate the object so that the fixed-point position is moved to the coordinate origin.
  2. Scale the object with respect to the coordinate origin.
  3. Translate the object so that the fixed point is returned to its original position.



(a) Original Position of Object and Fixed Point

(b) Translate Object so that Fixed Point (x_f, y_f) is at Origin

(c) Scale Object with Respect to Origin

(d) Translate Object so that the Fixed Point is Returned to Position (x_f, y_f)

- **General Two-Dimensional Fixed-Point Scaling**

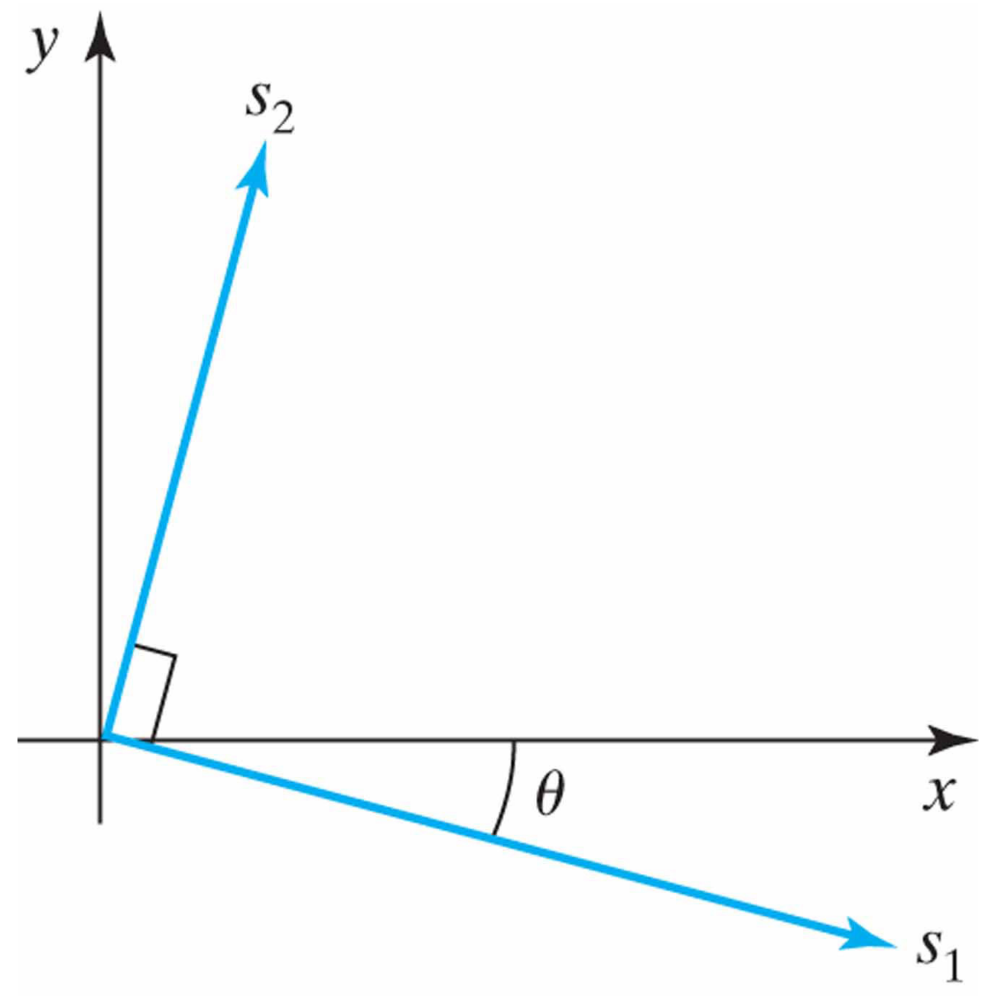
  - Concatenating the matrices for these three operations produces the required scaling matrix:

$$\begin{bmatrix} 1 & 0 & x_f \\ 0 & 1 & y_f \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -x_f \\ 0 & 1 & -y_f \\ 0 & 0 & 1 \end{bmatrix}$$
$$= \begin{bmatrix} s_x & 0 & x_f(1 - s_x) \\ 0 & s_y & y_f(1 - s_y) \\ 0 & 0 & 1 \end{bmatrix}$$

Or

$$\mathbf{T}(x_{\mathrm{f}}, y_{\mathrm{f}}) \cdot \mathbf{S}(s_x, s_y) \cdot \mathbf{T}(-x_{\mathrm{f}}, -y_{\mathrm{f}}) = \mathbf{S}(x_{\mathrm{f}}, y_{\mathrm{f}}, s_x, s_y)$$

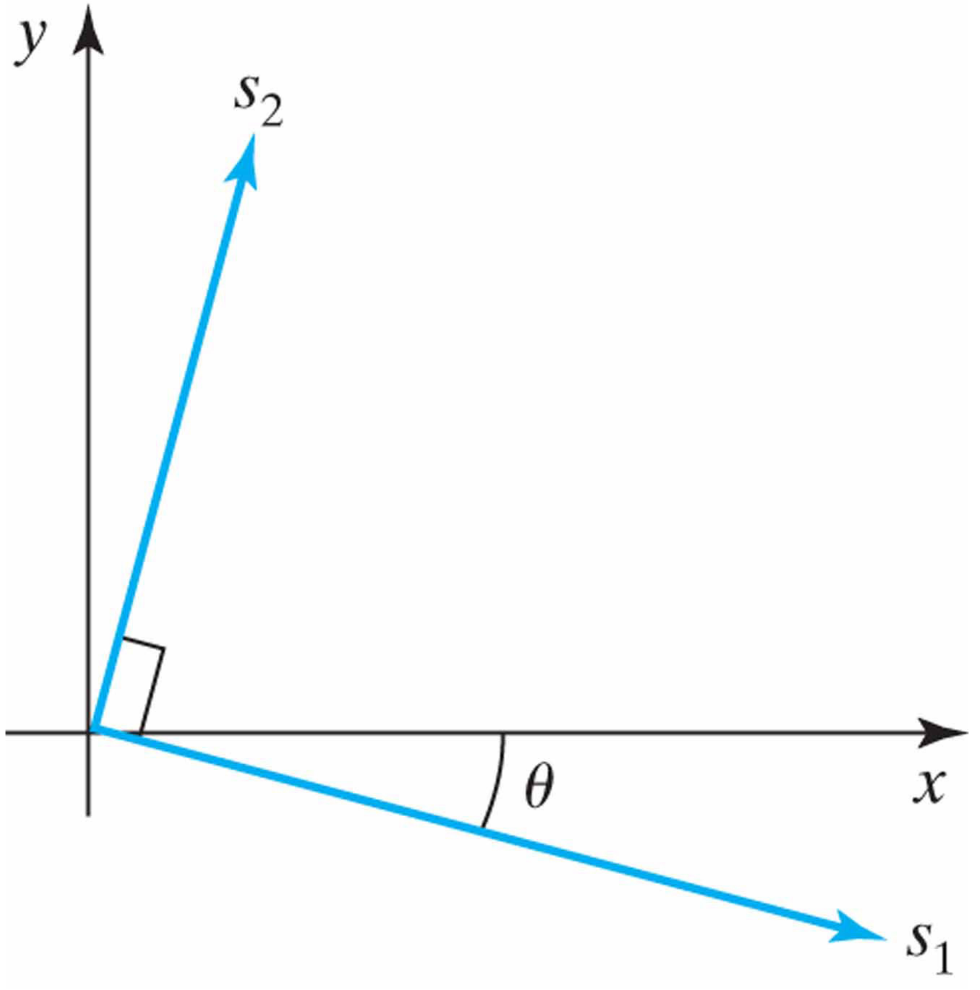
- **General Two-Dimensional Scaling Directions**
  - Parameters s_x and s_y scale objects along the x and y directions.
  - We can scale an object in other directions by rotating the object to align the desired scaling directions with the coordinate axes before applying the scaling transformation.



- **General Two-Dimensional Scaling Directions**

1. Perform a rotation so that the directions for s_1 and s_2 coincide with the x and y axes, respectively.
2. Scale the object with respect to the coordinate origin.
3. Opposite rotation to return points to their original orientations

$$\mathbf{R}^{-1}(\theta)\cdot\mathbf{S}(s_1,s_2)\cdot\mathbf{R}(\theta) = \begin{bmatrix} s_1\cos^2\theta + s_2\sin^2\theta & (s_2-s_1)\cos\theta\sin\theta & 0 \\ (s_2-s_1)\cos\theta\sin\theta & s_1\sin^2\theta + s_2\cos^2\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

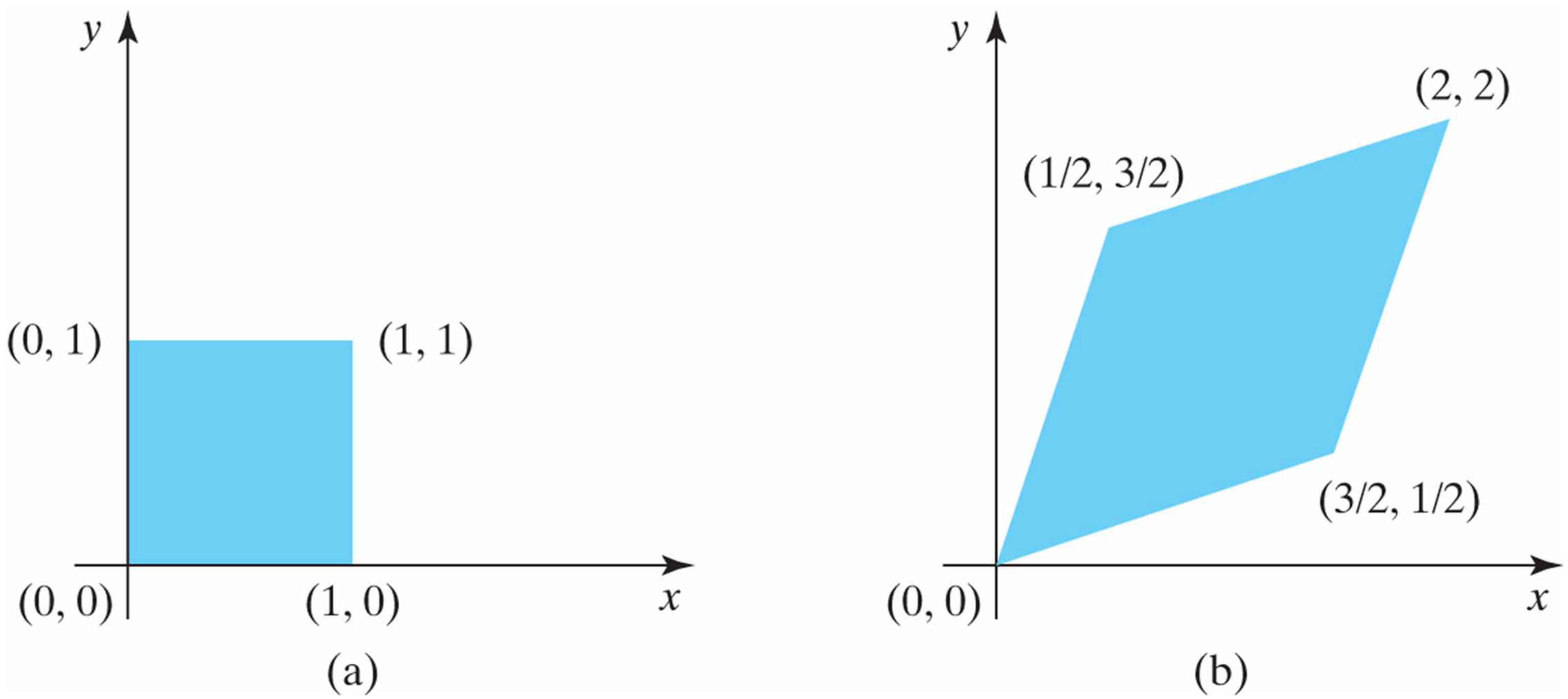


- # **General Two-Dimensional Scaling Directions**

Example

$$\mathbf{R}^{-1}(\theta)\cdot\mathbf{S}(s_1,s_2)\cdot\mathbf{R}(\theta)$$
$$=\begin{bmatrix} s_1\cos^2\theta+s_2\sin^2\theta & (s_2-s_1)\cos\theta\sin\theta & 0\\ (s_2-s_1)\cos\theta\sin\theta & s_1\sin^2\theta+s_2\cos^2\theta & 0\\ 0 & 0 & 1\end{bmatrix}$$

A square (a) is converted to a parallelogram (b) using the composite transformation matrix 7-39, with $s_1 = 1$, $s_2 = 2$, and $\theta = 45°$.



- **Matrix Concatenation Properties**

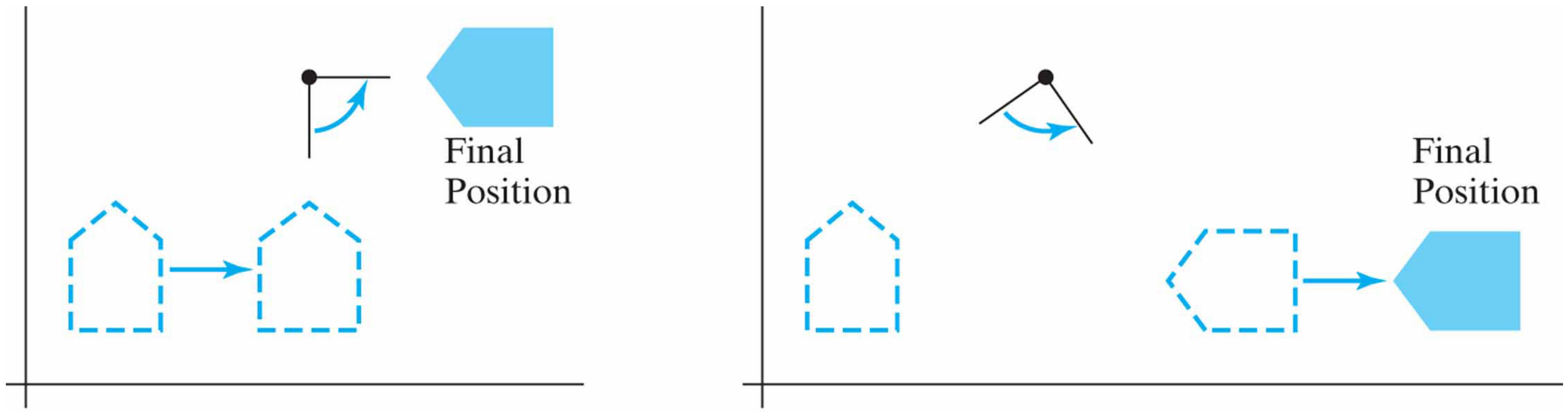
  - Multiplication of matrices is associative.

$$\mathbf{M}_3 \cdot \mathbf{M}_2 \cdot \mathbf{M}_1 = (\mathbf{M}_3 \cdot \mathbf{M}_2) \cdot \mathbf{M}_1 = \mathbf{M}_3 \cdot (\mathbf{M}_2 \cdot \mathbf{M}_1)$$

  - Transformation products may not be commutative.

$$\mathbf{M}_2 \cdot \mathbf{M}_1 \neq \mathbf{M}_1 \cdot \mathbf{M}_2$$

Reversing the order in which a sequence of transformations is performed may affect the transformed position of an object. In (a), an object is first translated in the x direction, then rotated counterclockwise through an angle of 45° . In (b), the object is first rotated 45° counterclockwise, then translated in the x direction.



(a) (b)

- **General Two-Dimensional Composite Transformations and Computational Efficiency**
  - A two-dimensional transformation, representing any combination of translations, rotations, and scalings, can be expressed as:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} rs_{xx} & rs_{xy} & trs_x \\ rs_{yx} & rs_{yy} & trs_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

  - The explicit calculations for the transformed coordinates are:

$$x' = x \cdot rs_{xx} + y \cdot rs_{xy} + trs_x$$
$$y' = x \cdot rs_{yx} + y \cdot rs_{yy} + trs_y$$

- **General Two-Dimensional Composite Transformations and Computational Efficiency**

Example

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} rs_{xx} & rs_{xy} & trs_x \\ rs_{yx} & rs_{yy} & trs_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- An object is to be scaled and rotated about its centroid coordinates (x_c, y_c) and then translated.

$$\mathbf{T}(t_x, t_\mathrm{y}) \cdot \mathbf{R}(x_c, y_\mathrm{c}, \theta) \cdot \mathbf{S}(x_c, y_\mathrm{c}, s_x, s_\mathrm{y})$$
$$= \begin{bmatrix} s_x \cos\theta & -s_y \sin\theta & x_c(1 - s_x\cos\theta) + y_c s_y \sin\theta + t_x \\ s_x \sin\theta & s_y\cos\theta & y_c(1 - s_y\cos\theta) - x_c s_x \sin\theta + t_y \\ 0 & 0 & 1 \end{bmatrix}$$

- **Two-Dimensional Rigid-Body Transformation**
  - If a transformation matrix includes only translation and rotation parameters, it is a **rigid-body transformation matrix.**
  - The general form for a two-dimensional rigid-body transformation matrix is:

$$\begin{bmatrix} r_{xx} & r_{xy} & tr_x \\ r_{yx} & r_{yy} & tr_y \\ 0 & 0 & 1 \end{bmatrix}$$

where the four elements r_{jk} *are the multiplicative rotation terms, and the elements* tr_x *and* tr_y *are the translational terms*

- **Two-Dimensional Rigid-Body Transformation**

$$\begin{bmatrix} r_{xx} & r_{xy} & tr_x \\ r_{yx} & r_{yy} & tr_y \\ 0 & 0 & 1 \end{bmatrix}$$

➢ Its upper-left 2×2 submatrix is an *orthogonal matrix.*

- *Each vector has unit length:*

$$r_{xx}{}^2 + r_{xy}{}^2 = r_{yx}{}^2 + r_{yy}{}^2 = 1$$

- *And the vectors are perpendicular*

$$r_{xx}r_{yx} + r_{xy}r_{yy} = 0$$

- **Two-Dimensional Rigid-Body Transformation**

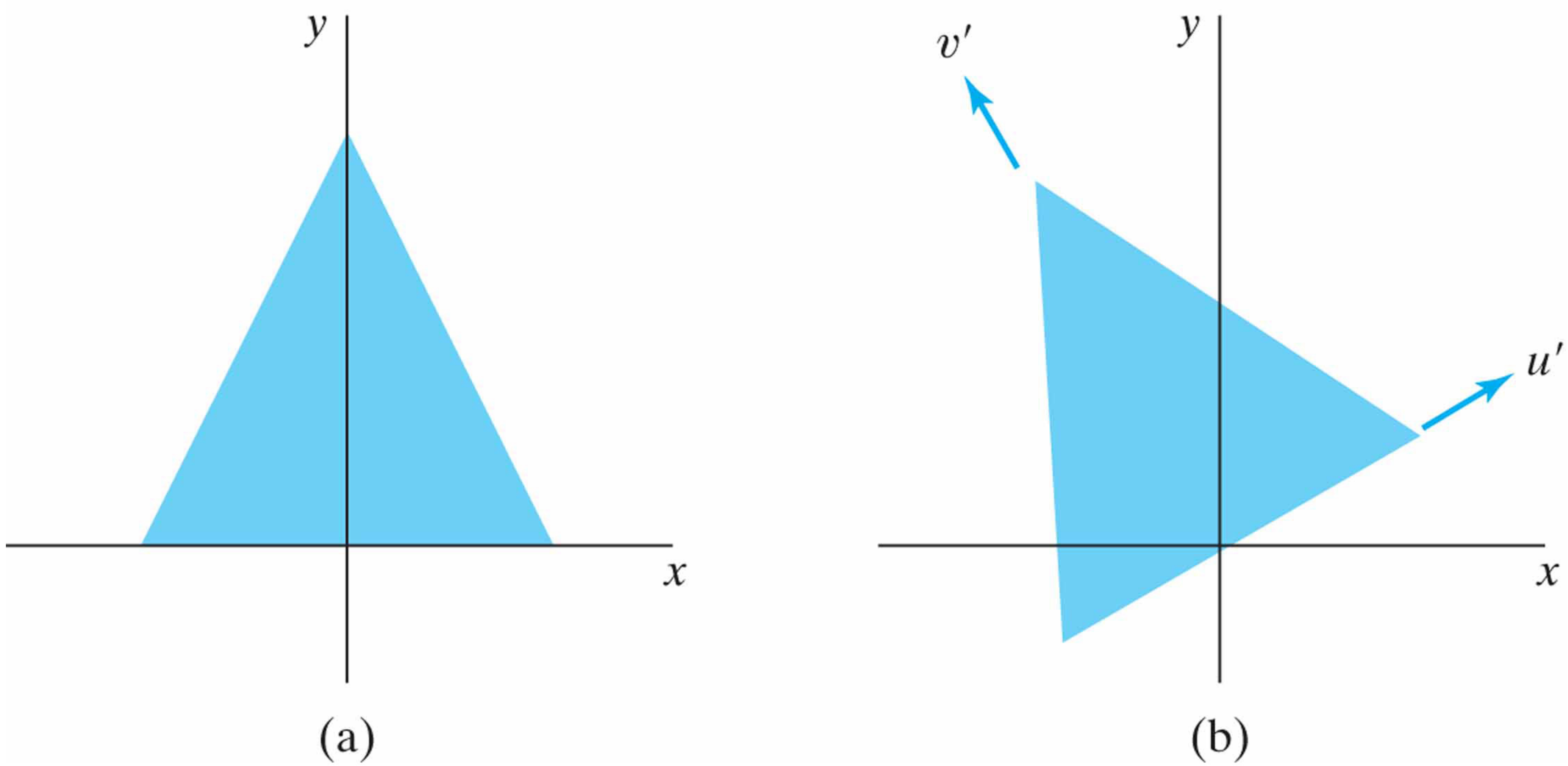
  - If these unit vectors are transformed by the rotation submatrix, then the vector (r_{xx}, r_{xy}) is converted to a unit vector along the x axis and the vector (r_{yx}, r_{yy}) is transformed into a unit vector along the y axis

$$\begin{bmatrix} r_{xx} & r_{xy} & 0 \\ r_{yx} & r_{yy} & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} r_{xx} \\ r_{xy} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} r_{xx} & r_{xy} & 0 \\ r_{yx} & r_{yy} & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} r_{yx} \\ r_{yy} \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

- **Constructing Two-Dimensional Rotation Matrices**
  - Obtain the transformation matrix within an object's local coordinate system when we know what its orientation is to be within the overall world-coordinate scene.

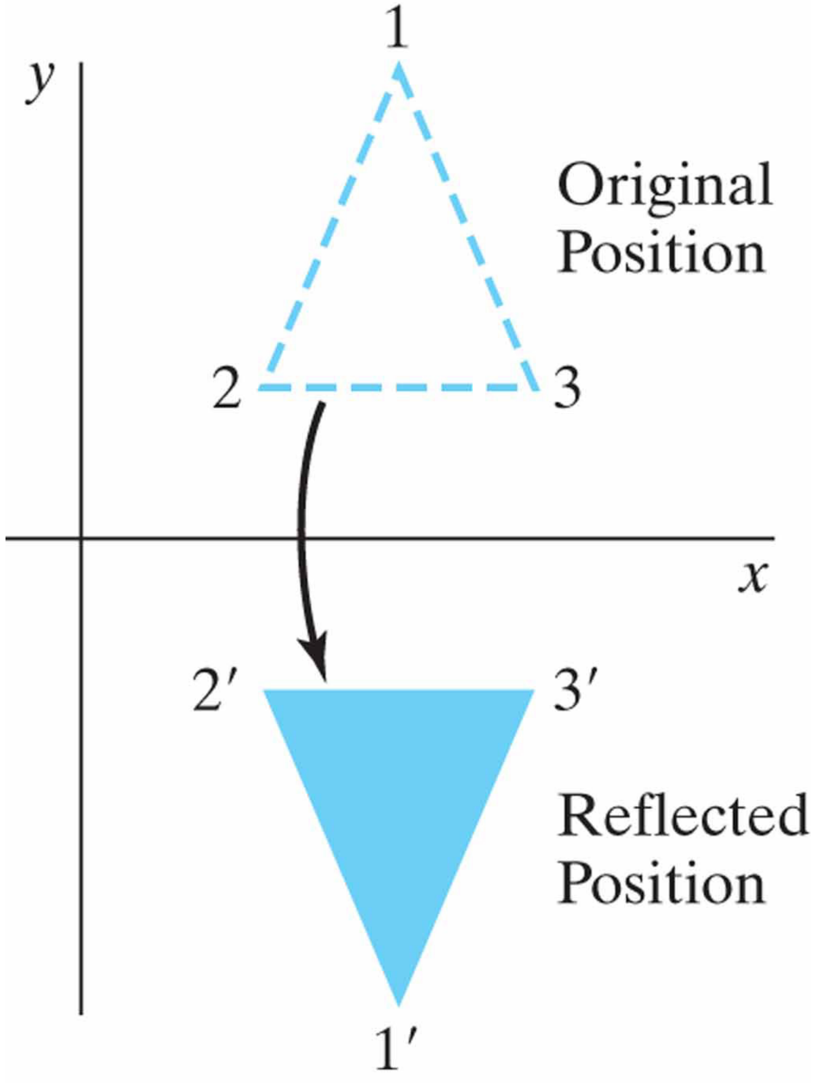
The rotation matrix for revolving an object from position (a) to position (b) can be constructed with the values of the unit orientation vectors u' and v' relative to the original orientation.



5.5 Other Two-Dimensional Transformations

- **Reflection**

- **Shear**

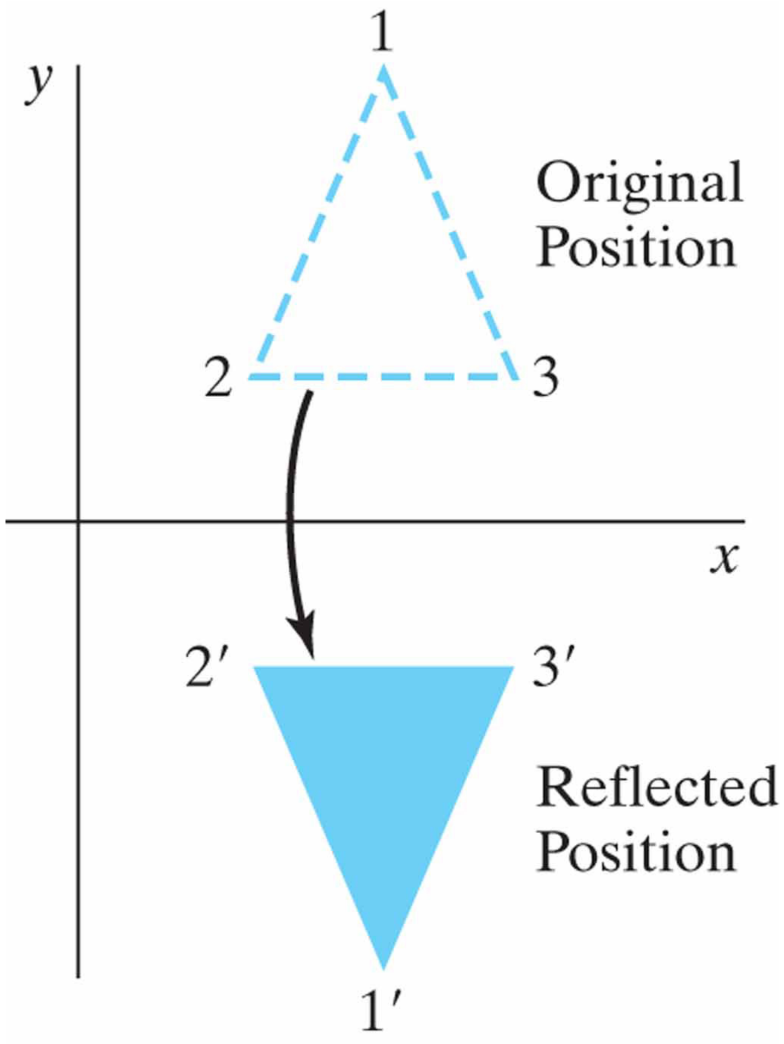


- **Reflection**

  - A transformation that produces a mirror image of an object is called a **reflection**.

    - *Reflection about the line y= 0 (the x axis) is accomplished with the transformation matrix*

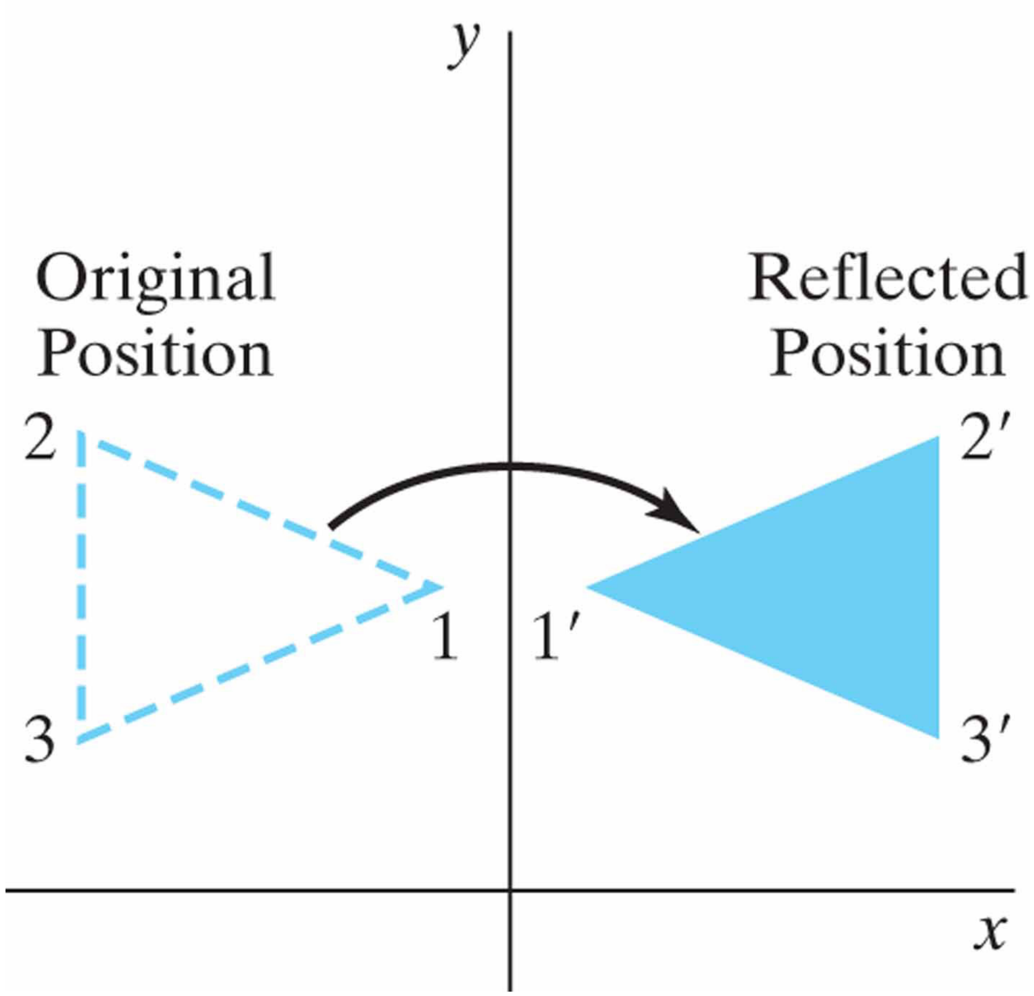
$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



- **Reflection**

  - *Reflection about the line x= 0 (the y axis) is accomplished with the transformation matrix*

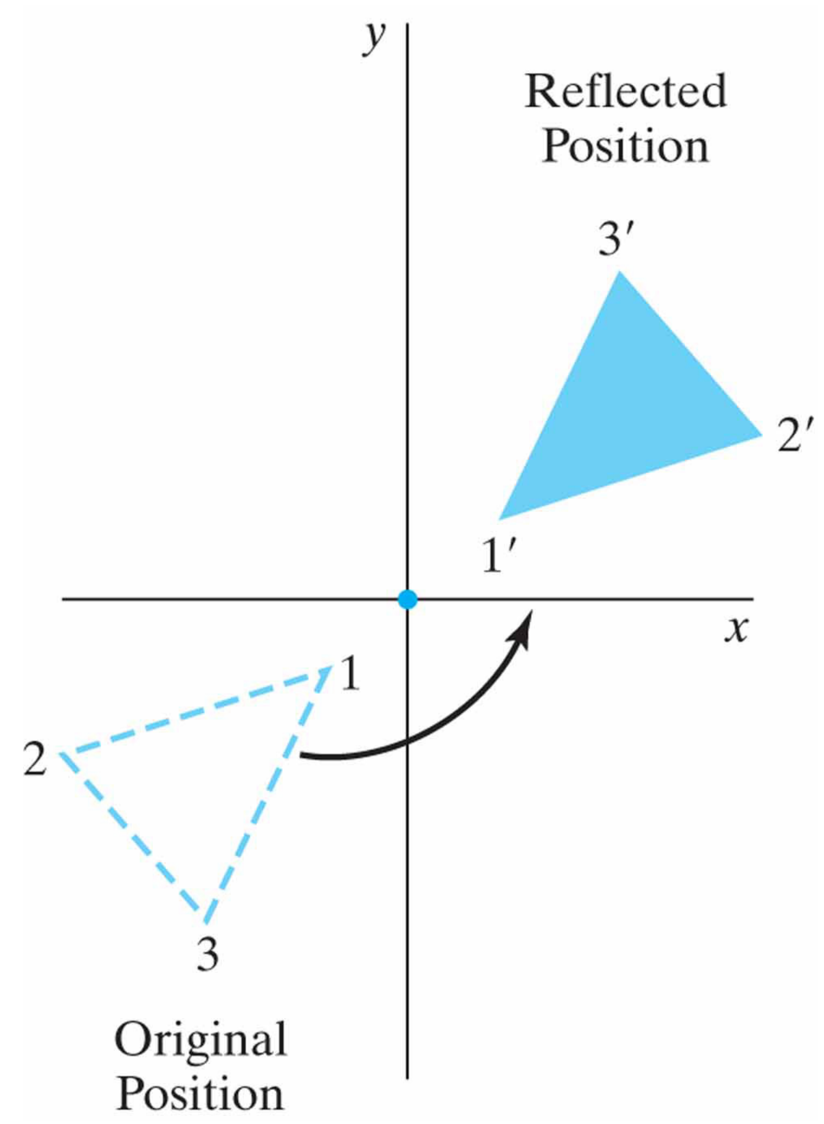
$$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



- **Reflection**

  - *Reflection relative to the coordinate origin.*
  - *It is equivalent to reflecting with respect to both coordinate axes*

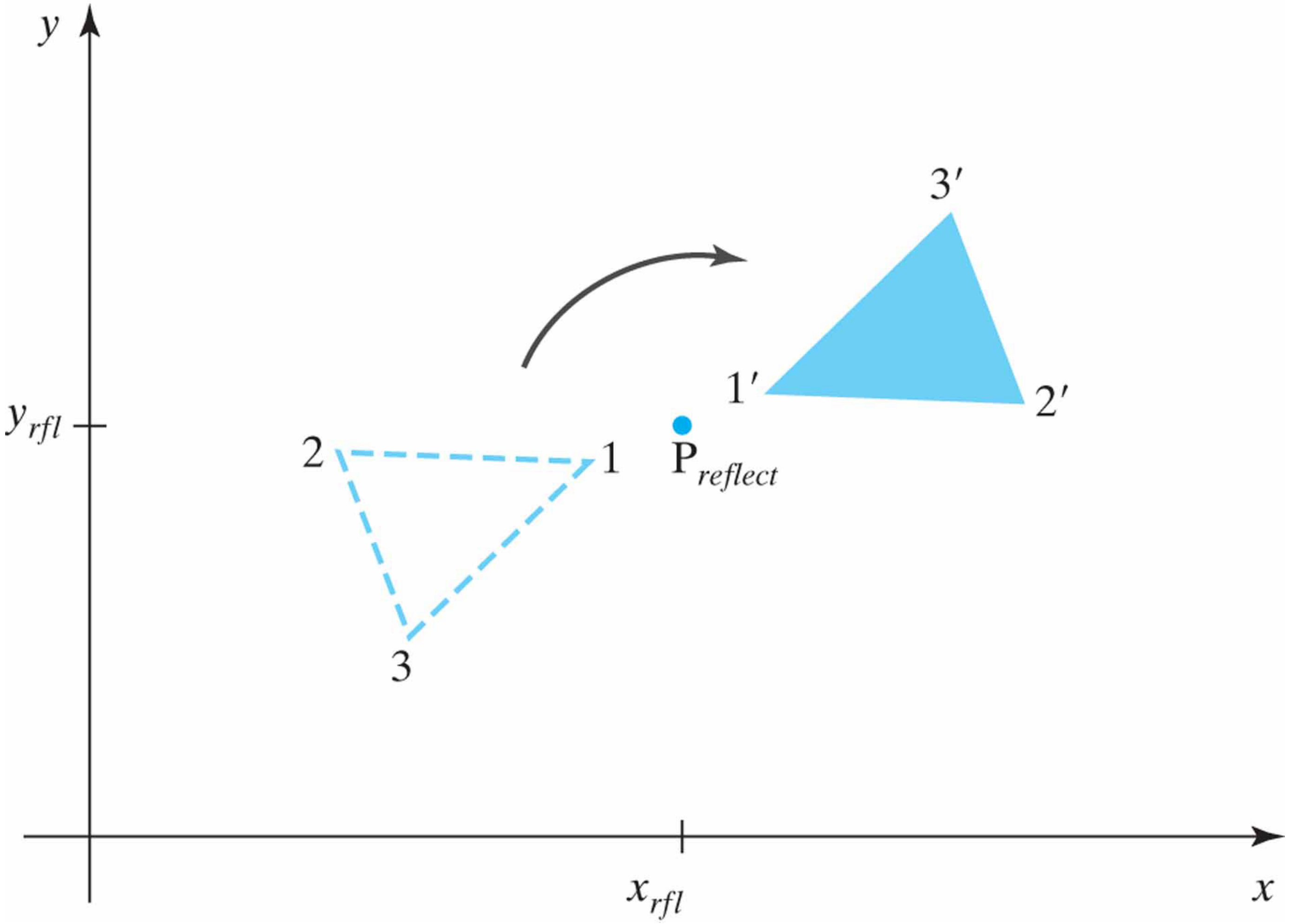
$$\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



- **Reflection**

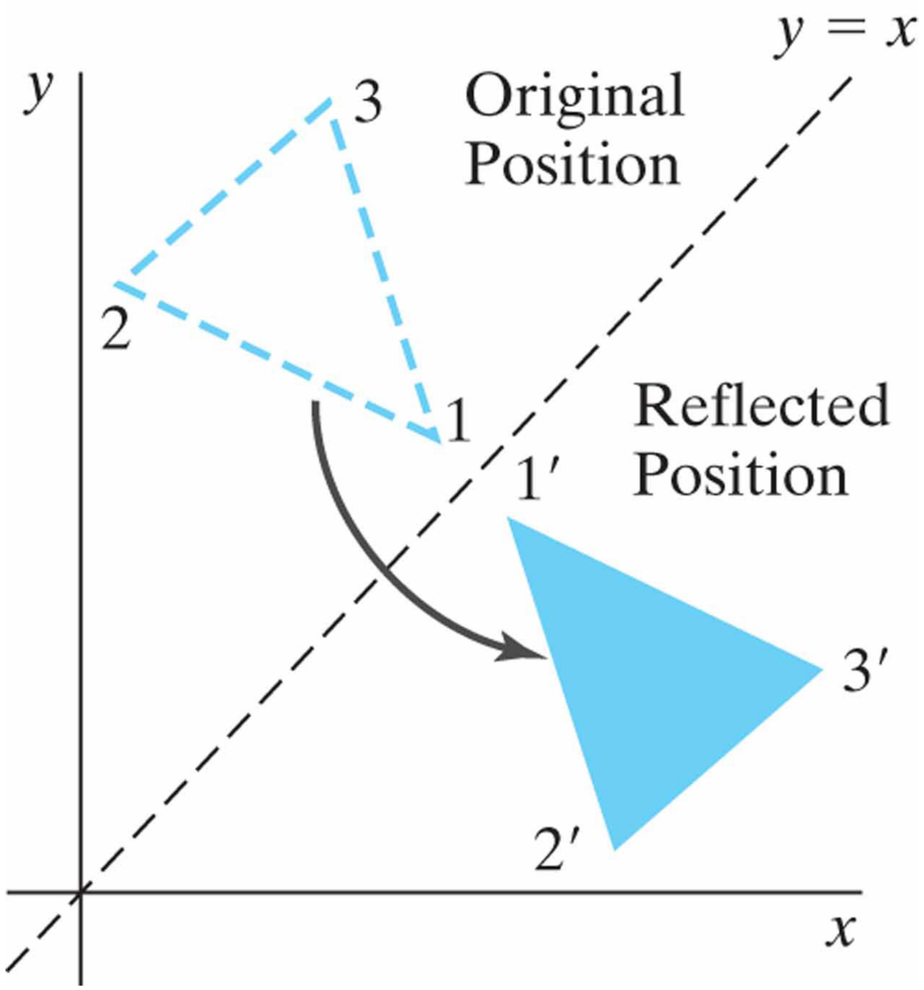
$$\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- This reflection is the same as a 180^0 rotation in the *xy* plane about the reflection point



- **Reflection**
  - If we choose the reflection axis as the diagonal line $y = x$, the reflection matrix is:

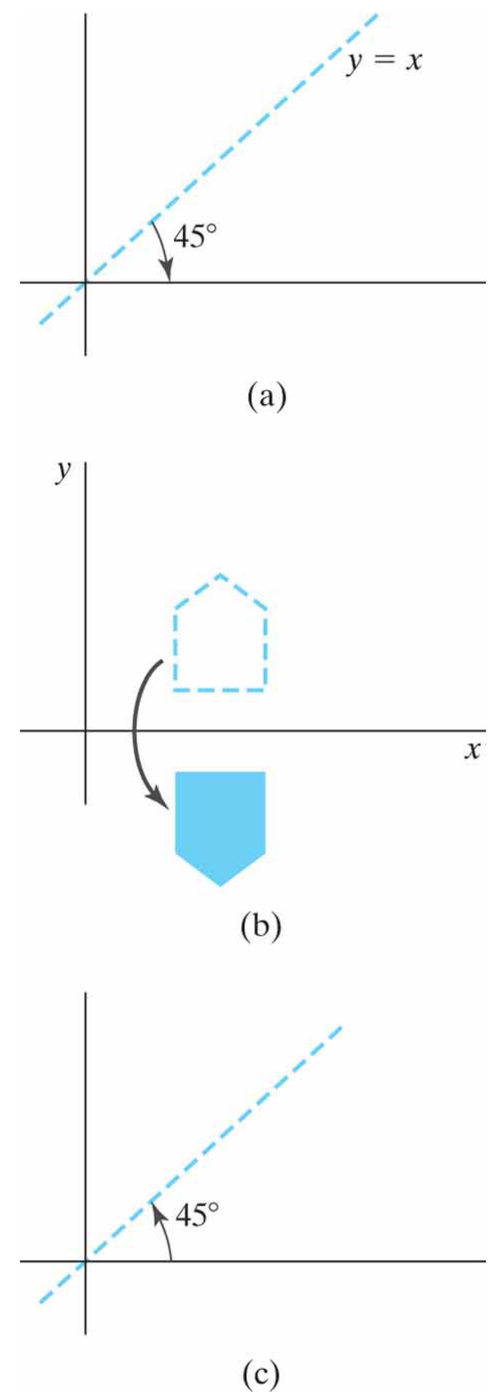
$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



- # **Reflection**

Sequence of transformations to produce a reflection about the line y = x: A clockwise rotation of 45° (a), a reflection about the x axis (b), and a counterclockwise rotation by 45° (c).

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



- We can derive this matrix by concatenating a sequence of rotation and coordinate axis reflection matrices

- **Shear**

  - A transformation that distorts the shape of an object such that the transformed shape appears as if the object were composed of internal layers that had been caused to slide over each other is called a **shear**.

  - Two common shearing transformations are those that shift coordinate x values and those that shift y values.

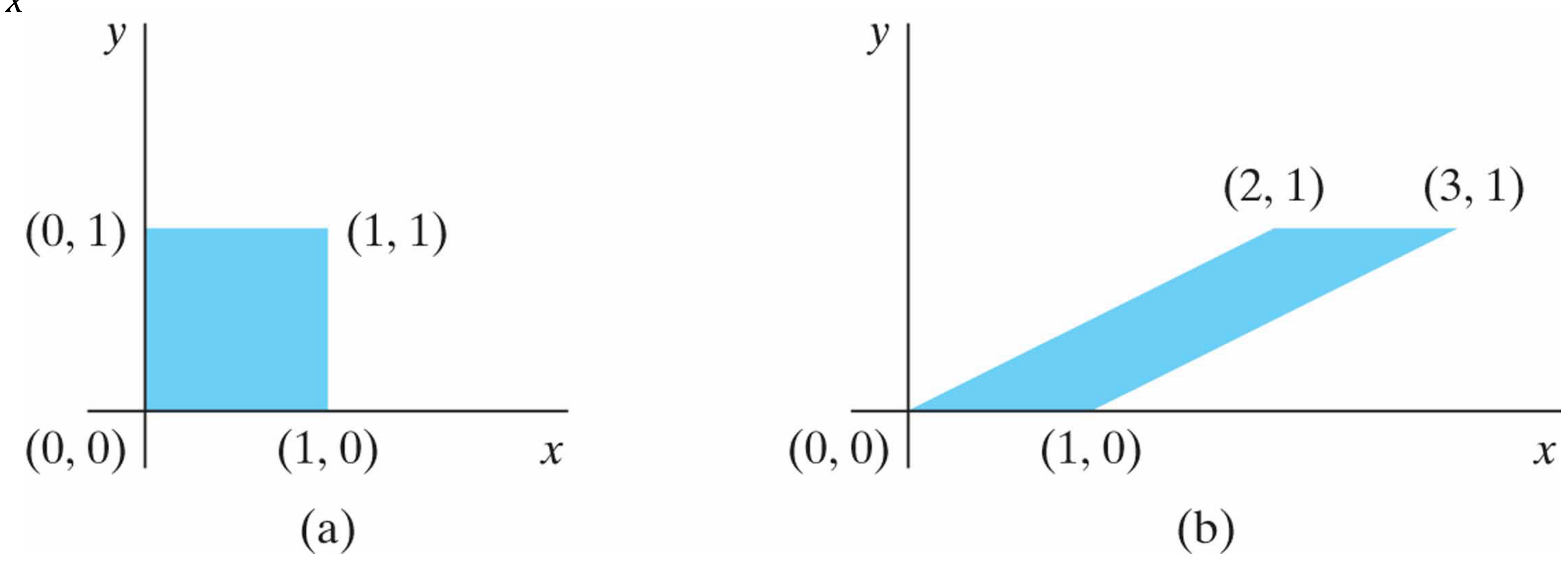
- **Shear**
  - An x-direction shear relative to the x axis is produced with the transformation matrix

$$\begin{bmatrix} 1 & sh_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Which transforms coordinate positions as

$$x' = x + sh_x \cdot y \qquad\qquad y' = y$$

A unit square (a) is converted to a parallelogram (b) using the x -direction shear matrix with $sh_x = 2$



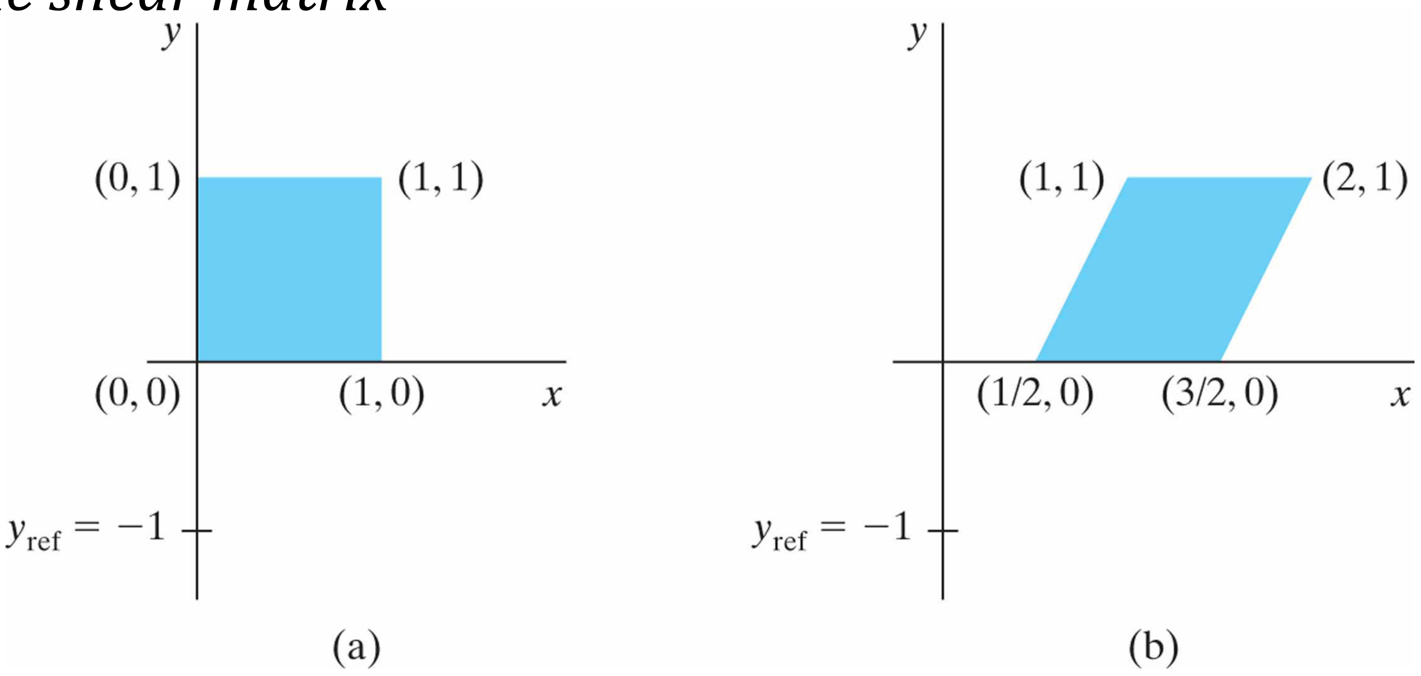
- **Shear**
  - We can generate x-direction shears relative to other reference lines with

$$\begin{bmatrix} 1 & sh_x & -sh_x \cdot y_{ref} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Now, coordinate positions are transformed as:

$$x' = x + sh_x \cdot (y - y_{ref}) \qquad y' = y$$

A unit square (a) is transformed to a shifted parallelogram (b) with $sh_x = 0.5$ and $y_{ref} = -1$ in the shear matrix



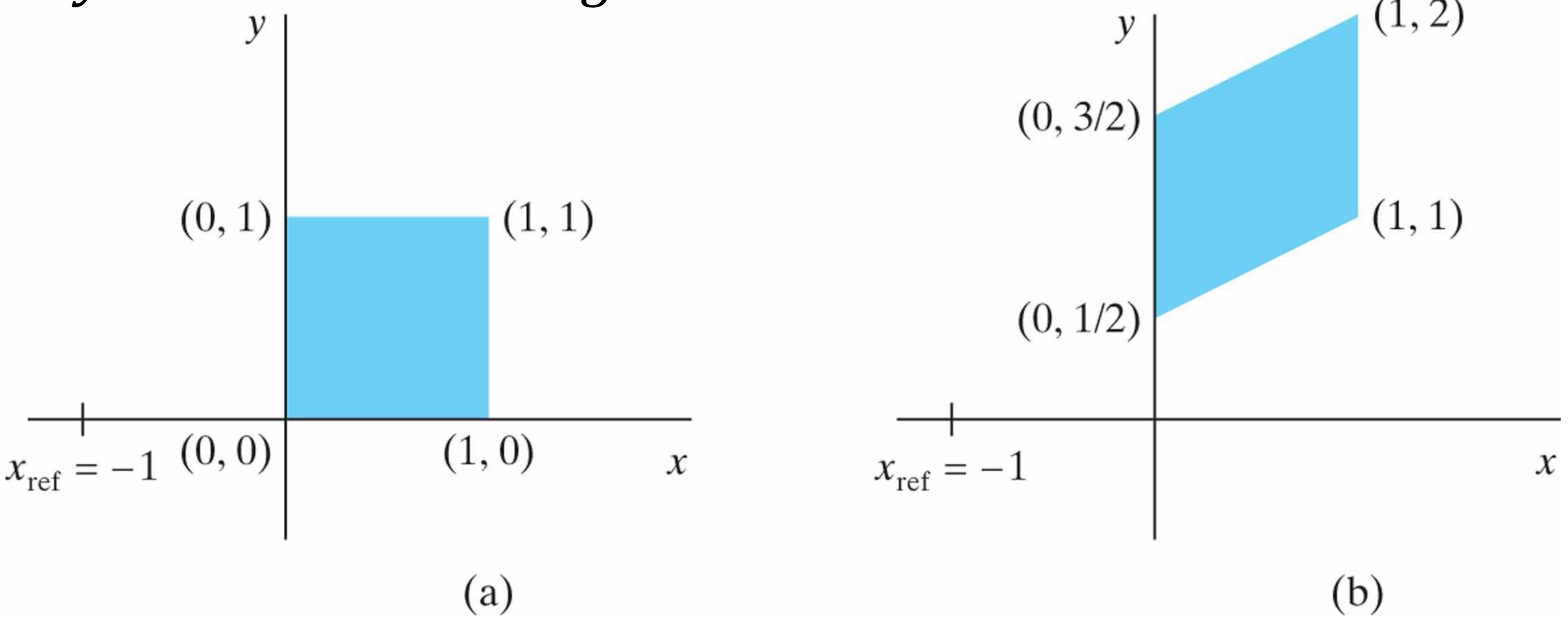
- **Shear**
  - A y-direction shear relative to the line $x = x_{ref}$ is generated with the transformation matrix

$$\begin{bmatrix} 1 & 0 & 0 \\ sh_y & 1 & -sh_y \cdot x_{ref} \\ 0 & 0 & 1 \end{bmatrix}$$

Which generates the transformed coordinate values

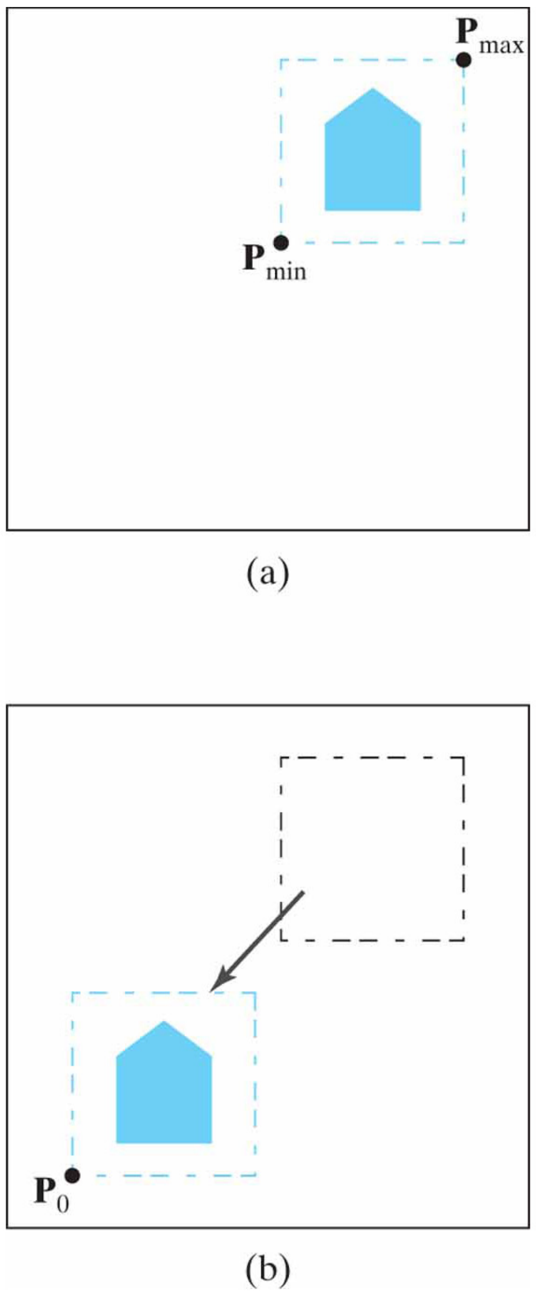
$$x' = x \qquad\qquad y' = y + sh_y(x - x_{ref})$$

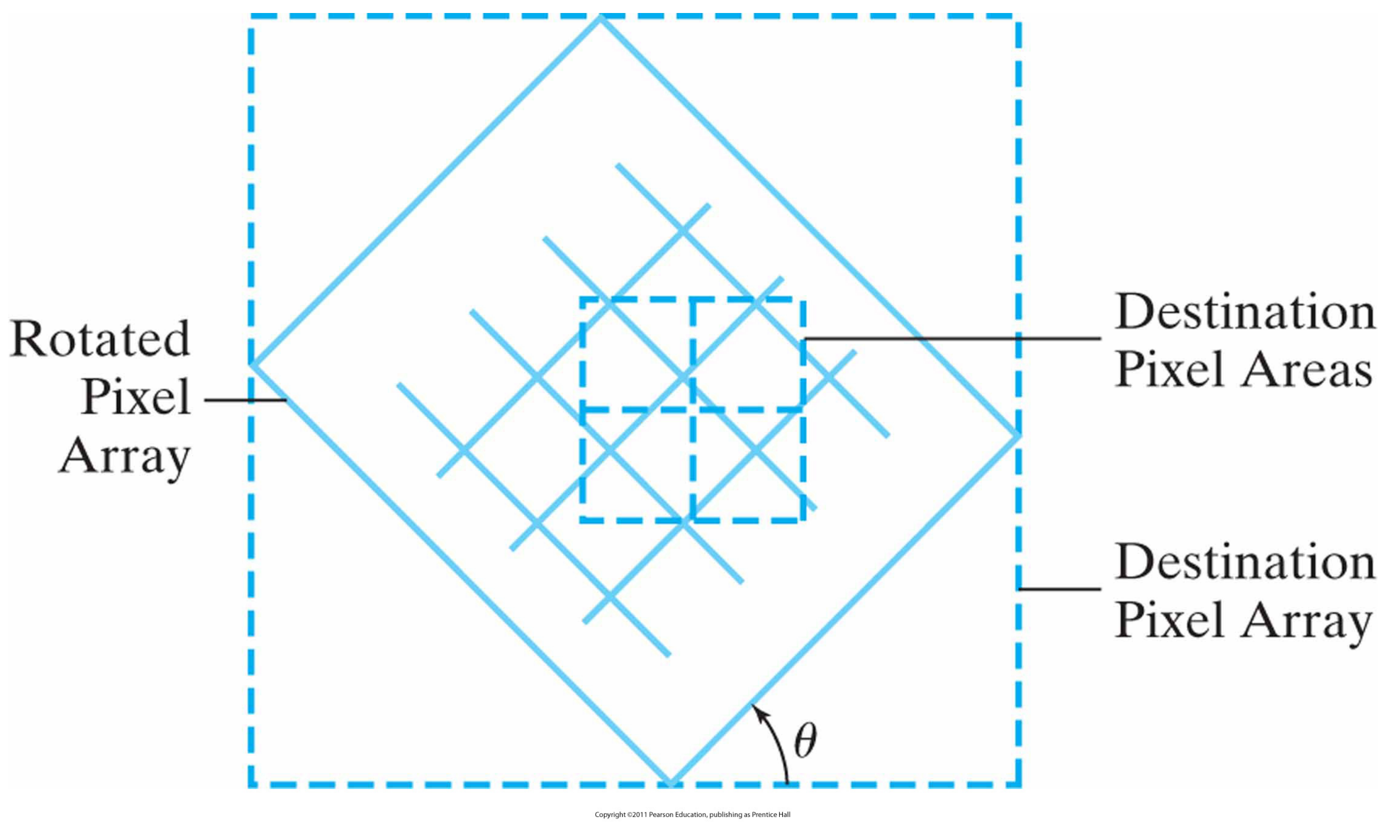
A unit square (a) is turned into a shifted parallelogram (b) with parameter values $sh_y = 0.5$ and $x_{ref} = -1$ in the y -direction shearing



5.6 Raster Methods for Geometric Transformations

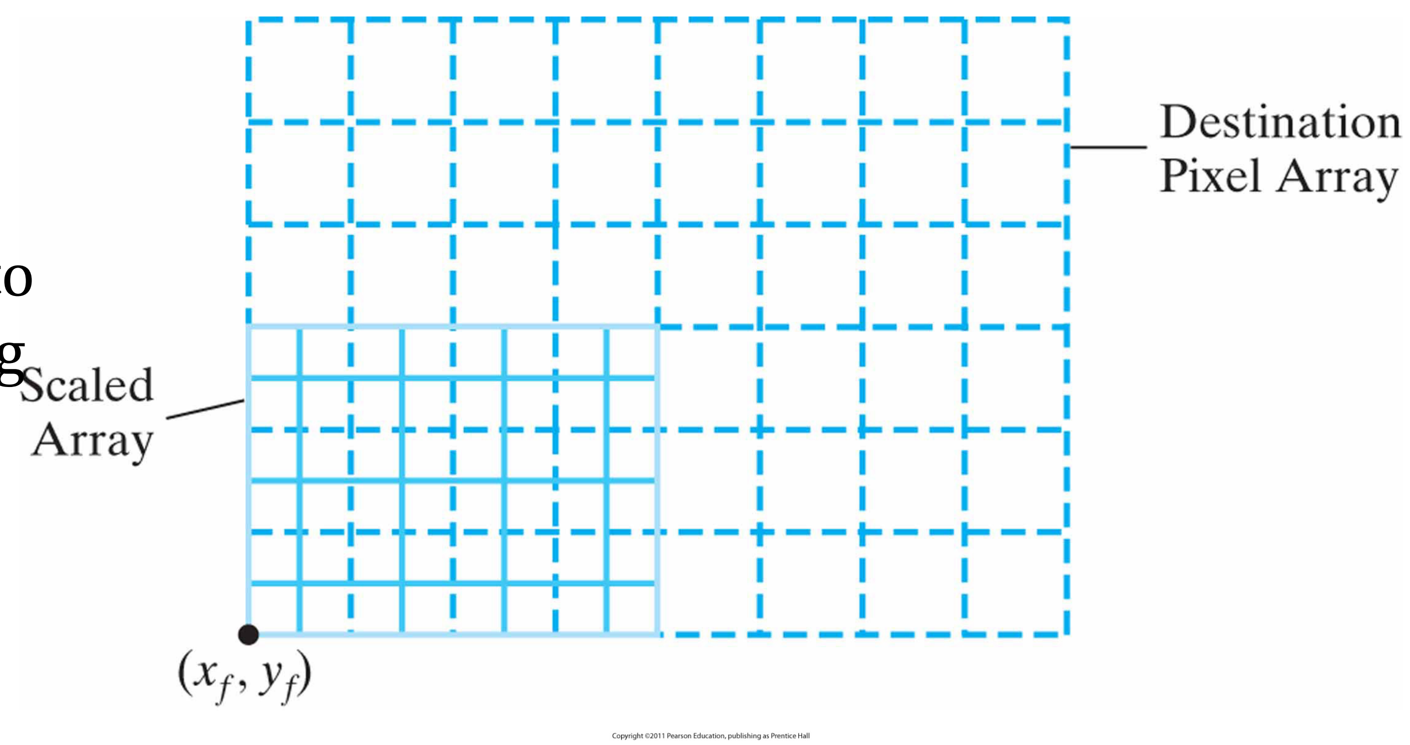
- Functions that manipulate rectangular pixel arrays are called *raster operations*, and moving a block of pixel values from one position to another is termed a *block transfer*, a *bitblt*, or a *pixblt*.





- *A raster rotation for a rectangular block of pixels can be accomplished by mapping the destination pixel areas onto the rotated block.*

- Mapping destination pixel areas onto a scaled array of pixel values. Scaling factors $s_x = s_y = 0.5$ are applied relative to fixed point (x_f, y_f).

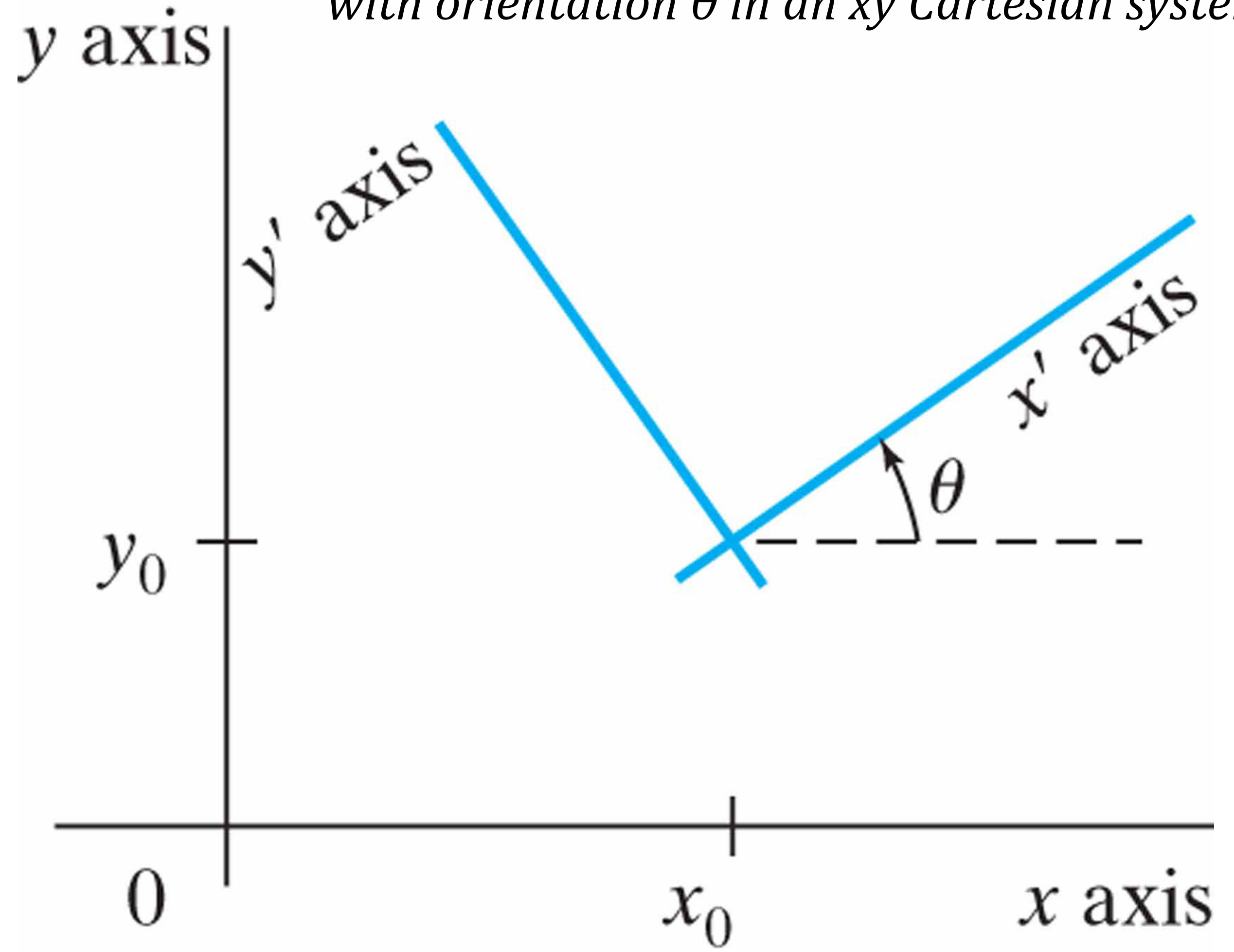


5.6 Transformation between Two-Dimensional Coordinate Systems

- Computer graphics applications involve coordinate transformations from one reference frame to another during various stages of scene processing.

1. Translate so that the origin *(x_0, y_0) of the x'y' system is moved to the origin (0,0) of the xy system*.
2. Rotate the x' axis onto the x axis.

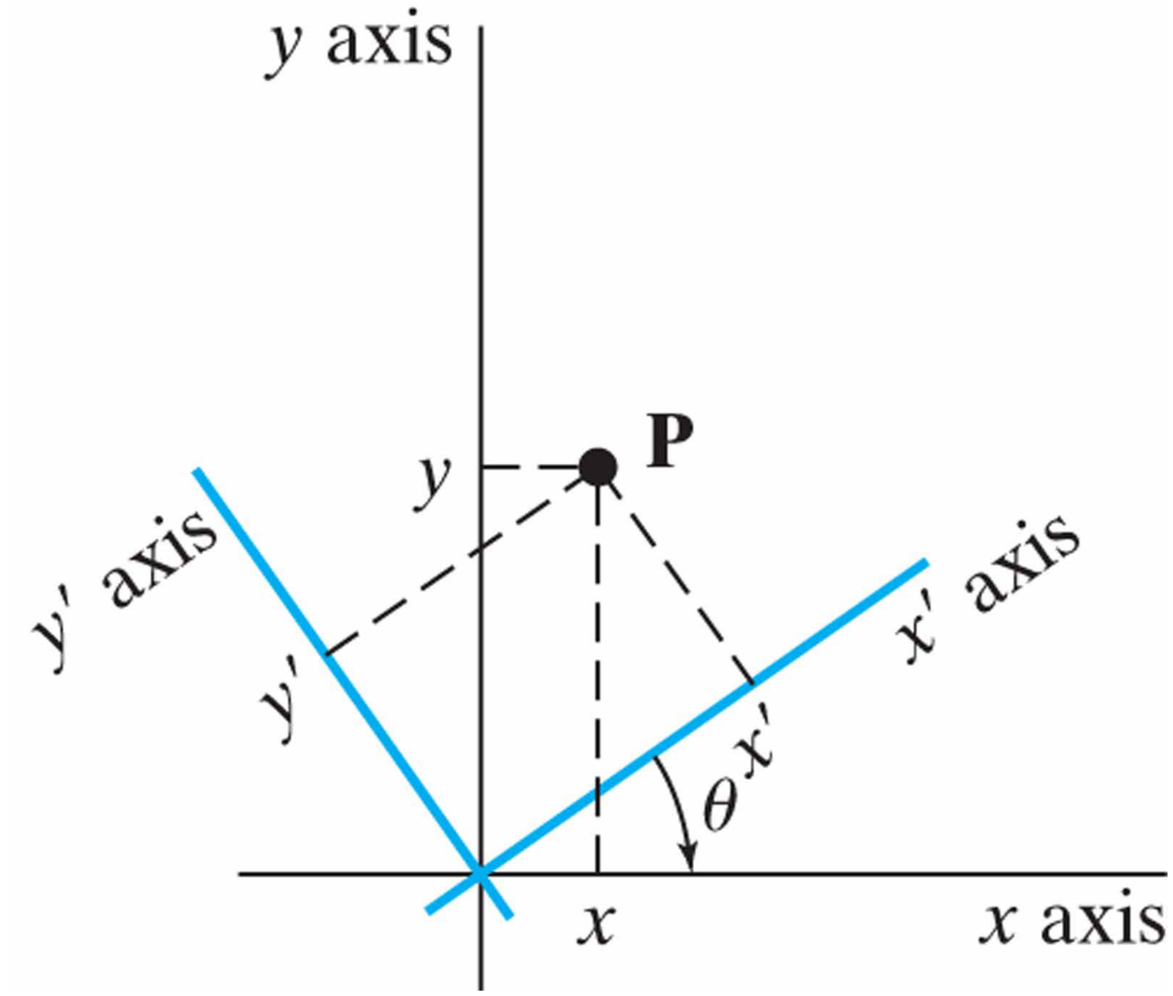
A Cartesian x' y' system positioned at (x_0, y_0) with orientation θ in an xy Cartesian system



- **Translation**

$$\mathbf{T}(-x_0,-y_0)=\begin{bmatrix}1 & 0 & -x_0\\ 0 & 1 & -y_0\\ 0 & 0 & 1\end{bmatrix}$$

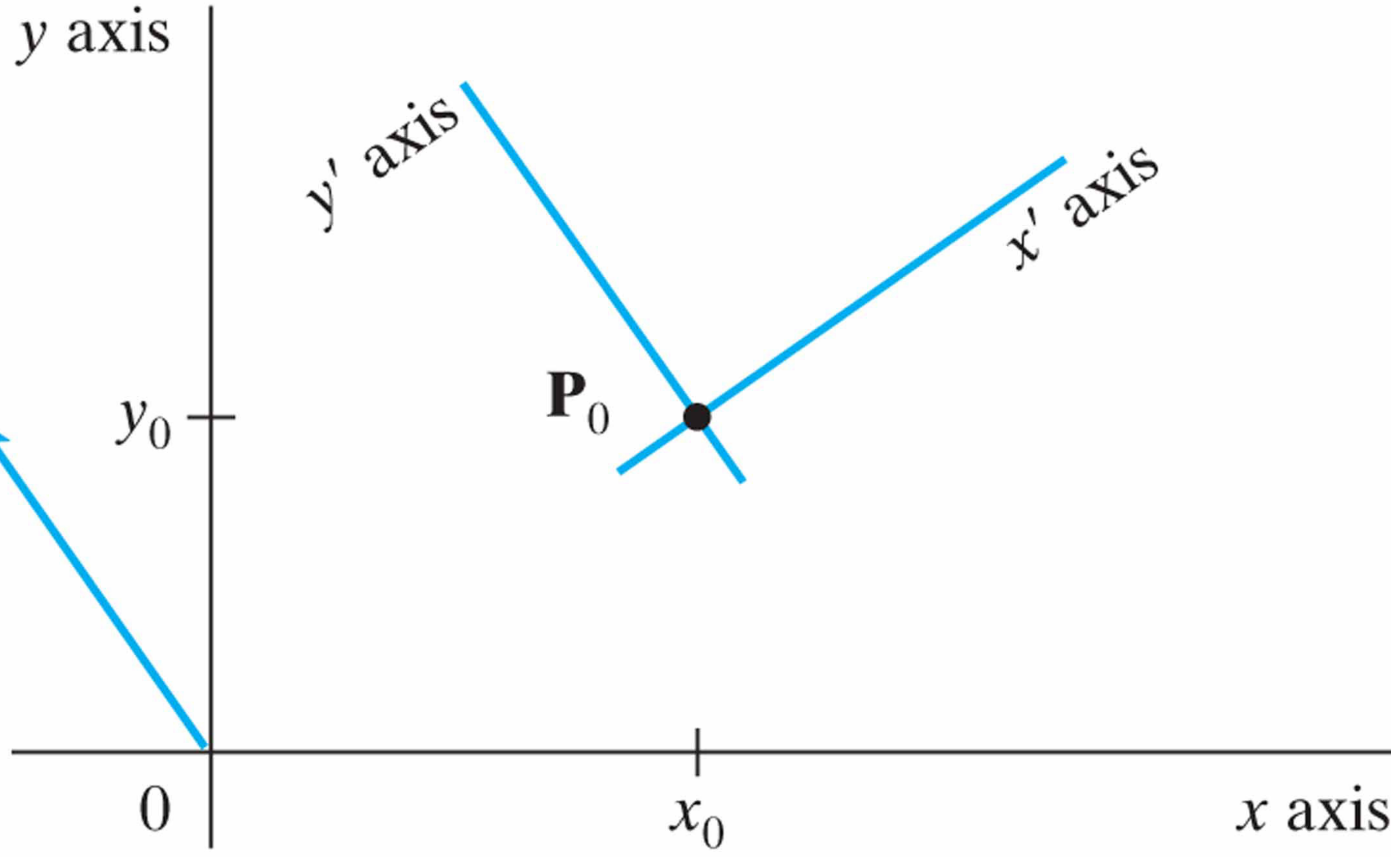
$$\mathbf{R}(-\theta)=\begin{bmatrix}\cos\theta & \sin\theta & 0\\ -\sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix}$$



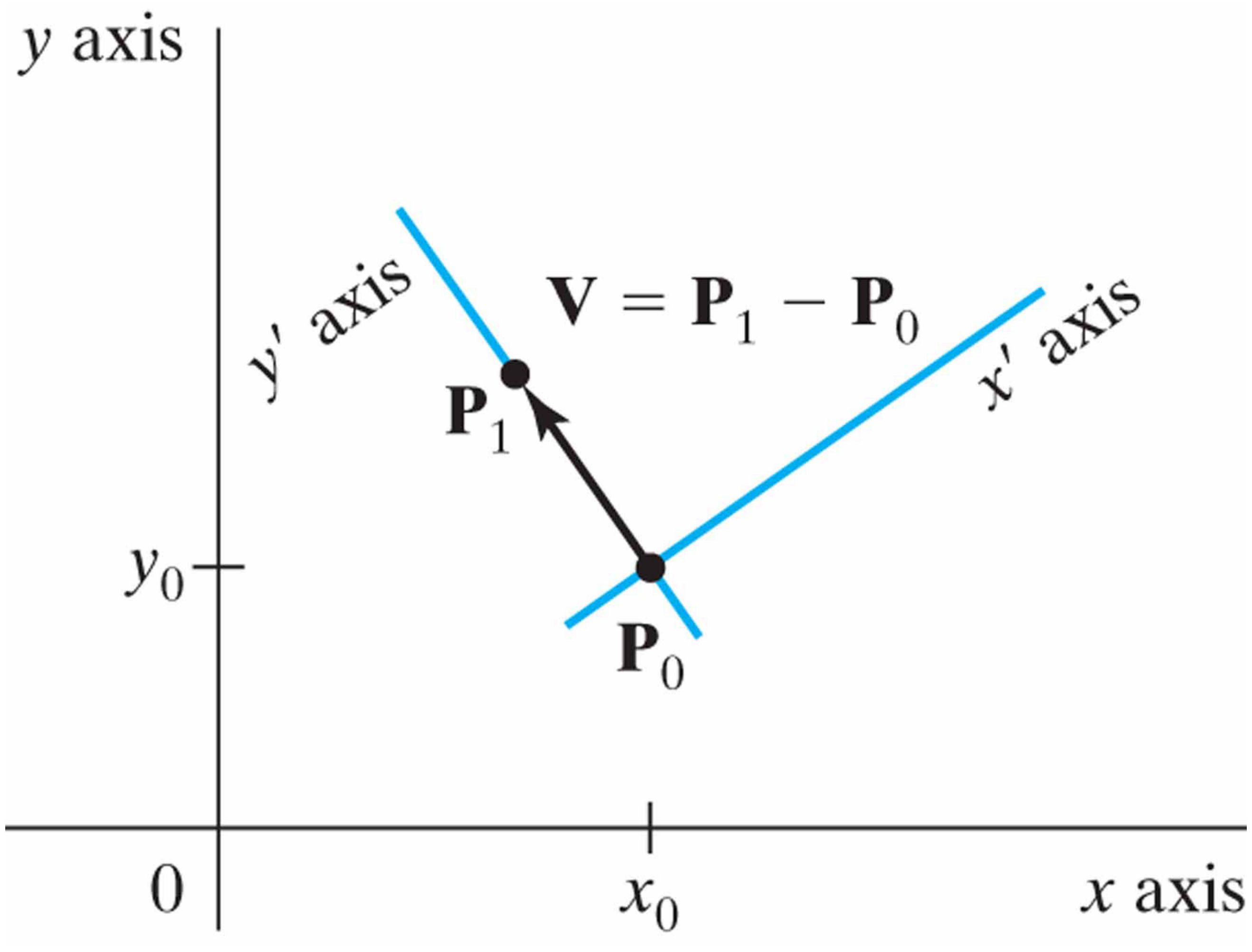
$$\mathbf{M}_{xy,x'y'}=\mathbf{R}(-\theta)\cdot\mathbf{T}(-x_0,-y_0)$$

- **Translation**
  - Another method for describing the orientation of the x'y' coordinate system is to specify a vector **V** that indicates the direction for the positive y' axis.

Cartesian system x' y' with origin at P0 = (x_0, y_0) and y' axis parallel to vector ***V***



- **Translation**
  - We can choose **V** relative to position $\mathbf{P}_0$



$$\mathbf{V} = \frac{\mathbf{P}_1 - \mathbf{P}_0}{|\mathbf{P}_1 - \mathbf{P}_0|}$$